2002

Synthesizing a Safety Controller for ConnectedSpaces Using Supervisory Control

Baskar Sridharan

Aditya P. Mathur
Purdue University, apm@cs.purdue.edu

Kai-Yuan Cai

Report Number:
02-023

TR # 02-023

Synthesis of a Safety Controller for ConnectedSpaces Using Supervisory Control

Baskar Sridharan[*], *Aditya P. Mathur*[†]
Software Engineering Research Center,
Dept. of Computer Science,
Purdue University,
West Lafayette, IN, USA
{baskars, apm}@cs.purdue.edu

Kai-Yuan Cai[‡]
Dept. of Automatic Control,
Beijing University of Aeronautics
and Astronautics,
Beijing 100083, China
kyc@ns.dept3.buaa.edu.cn

Keywords: Safety-critical systems, Real-time systems,
Discrete event systems, Supervisory control, and SmartHomes

Abstract

A collection of one or more devices, each described by its Digital Device Manual and reachable over a network, is a ConnectedSpace. The behavior of each device is expressed using an extended finite state machine. A set of policies may be enforced on the ConnectedSpace to ensure its safe behavior. Such safety policies are monitored and enforced by a safety controller. We show how a safety controller is synthesized by expressing the ConnectedSpace and the safety policies using the Timed Transition Model (TTM) and the Real-Time Temporal Logic (RTTL), respectively. The resulting behavior of the ConnectedSpace, enforced by the safety controller, is guaranteed to satisfy the safety policies. The notions of policy relaxation and safety ranking are novel to this work. The synthesis procedure is modified to incorporate these two notions. We present an experimental evaluation of the modified synthesis procedure. The experimental results show that the synthesis procedure is scalable with respect to the number of devices, number of policies, and the number of states of the finite state machines.

1 Introduction

The availability of low cost and powerful integrated circuits coupled with the widespread access to the Internet is fueling the development of a variety of "smart" devices. Such devices are capable of being monitored and controlled remotely, via the Internet or a local network. The devices also carry information that describe their attributes, functions, and behavior. A collection of such devices is a ConnectedSpace. The commercial domains in which ConnectedSpaces are deployed include hospitals, aircraft, health-care units, automotives, and homes. Ensuring the safety of such environments in the presence of these devices is critical. In such environments, interactions of two or more devices may lead to an unsafe situation. For example, consider a ConnectedSpace deployed in the cabin of a passenger aircraft. NASA's Air Safety Reporting System lists 52 incidents where a portable electronic device was suspected to interfere with the aircraft navigation and control systems [5, 10]. For example, the headphones used by a passenger on a Boeing 757 was suspected to have caused all three autopilot systems on the aircraft to stop functioning. The use of a cellular phone inside a Cessna 340/A was reported as the likely cause for erroneous readings of the cockpit meters. Similar

[*]Baskar Sridharan's work was supported in part by SERC.
[†]Aditya Mathur's work was supported in part by SERC and NSF.
[‡]Kai-Yuan Cai's work was supported by the National Natural Science Foundation of China and the "863" programme of China.

examples may be found in hospital and health-care environments. For example, the interference of an Magnetic Resonance Imaging (MRI) device with a cardiac pacemaker is well known [20]. Such interference may even be fatal [6]. It is possible to prevent such interactions by specifying and enforcing *safety policies*. The safety policies specify rules that guarantee a safe behavior of the devices in the environment under consideration.

The ability to monitor and control the devices from an external source provides opportunities for automatic control of the safety of the ConnectedSpace, and hence the environment. The safety policies, such as those listed in [16, 1, 19, 3, 17, 18], may be used for the control. The ConnectedSpace may be viewed as a Real-Time Discrete Event System (RDES). Control-theoretic concepts for RDES may then be used for synthesizing the safety controller for ConnectedSpaces.

The theory of supervisory control of Discrete Event Systems (DES), based on a framework of automata and formal languages, was proposed by Ramadge and Wonham [15, 14]. This theory (RW theory) may be used to synthesize a controller when the DES and the control specifications are both modeled as generators of formal languages. The controller enforces the specifications on the DES. Ostroff [11, 12] proposed a framework based on Timed Transition Model (TTM) and Real-Time Temporal Logic (RTTL) for synthesizing controllers for RDES. Given an RDES modeled using TTM, and the control specifications expressed using RTTL, the controller may be synthesized using the Controller Design Procedure (CDP) described in [11]. It is important to emphasize that these techniques employ a *constructive* approach in contrast to a *verification-based* approach for synthesizing controllers. A constructive approach aims to synthesize a controller that is correct with respect to a given criteria. In a verification-based approach, the controller is synthesized and then verified for its correctness.

In this work we describe a constructive approach for synthesizing safety controllers for ConnectedSpaces, using RTM/RTTL framework. The key contributions of this work are: (a) an approach to express ConnectedSpaces and their safety policies in the TTM/RTTL framework, (b) the notions of safety criticality rank and policy relaxation duration, (c) modifications to CDP to accommodate safety criticality rank and policy relaxation duration, and (d) an experimental evaluation of the proposed approach.

The remainder of the paper is organized as follows. ConnectedSpaces and Digital Device Manuals are described in Section 2. Safety policies, safety criticality rank, and policy relaxation duration are described in Section 3. Section 4 and Section 5 show how ConnectedSpaces and safety policies can be expressed using TTM and RTTL, respectively. A modified CDP and its application in synthesizing a safety controller are proposed in Section 6. The results of an experimental evaluation of the modified CDP is presented in Section 7. Possible extensions of the work are discussed in Section 8.

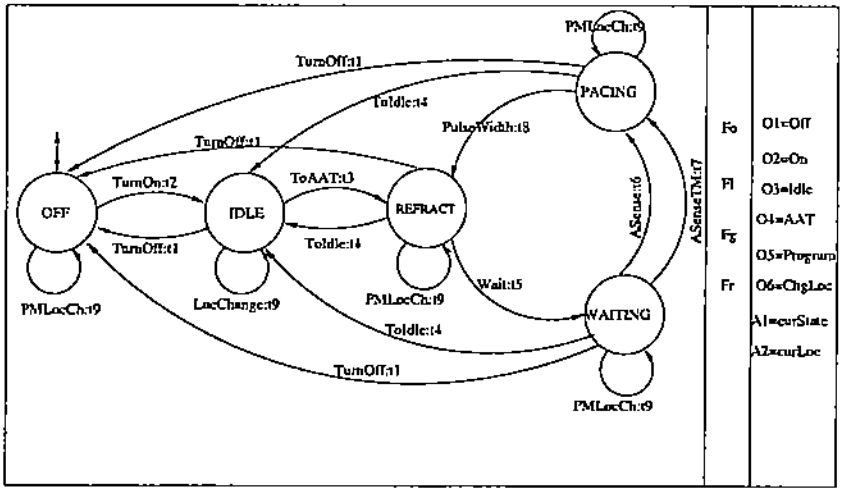
2 ConnectedSpace and Digital Device Manuals

A ConnectedSpace is a collection of one or more devices reachable across a network. The behavior, functions, and attributes of a device are specified as a Digital Device Manual (DDM) [21]. A device may carry either its DDM or a pointer to the location of its DDM.

Formally, a ConnectedSpace is defined as $CS = \{D_1, D_2, \ldots, D_m\}$ where each D_i is a DDM. A DDM D is given by (CSM, F, LS, O, A) where

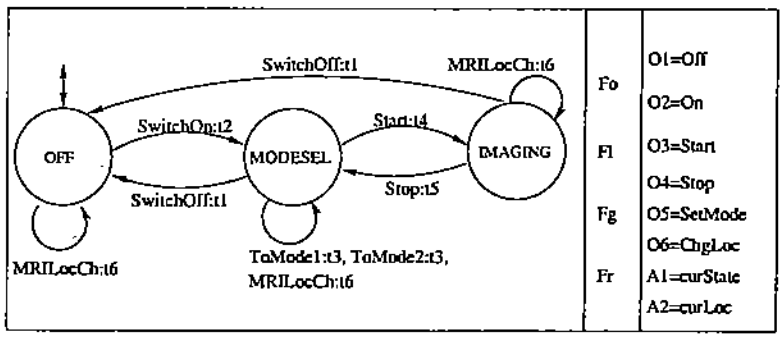
- CSM is an augmented finite state automaton for the device. CSM is given by a 6-tuple $(Q, \Sigma, \delta, \Omega, q_0, SD)$ where $Q = Q_u \cup Q_s$ is a finite set of states, Σ is a possibly empty set of events, $\delta : Q \times \Sigma \rightarrow Q$ is the transition function, $\Omega : \Sigma \rightarrow \mathbb{N}$ gives the cost associated with a transition, q_0 is the initial state of the device, and $SD : Q \rightarrow string$ is the *semantic description* function. Q_s is a non-empty set of states. Q_u is a non-empty set of states such that $\forall s \in Q_u$, there is a $\tau \in \Sigma$ such that $\delta(s, \tau) \in Q_s$. In practice, Q_u are the set of states that may potentially lead to an unsafe situation of CS. Q_s is the set of states that are guaranteed not to lead to any unsafe situation. For the purpose of this paper, we assume that Ω gives the maximum time duration required to complete a transition.

- $F = \{f_0, f_1, f_2, \ldots, f_k\}$ is a set of mapping functions. $f_i, i = 1 \ldots k$ are functions defined by the device.



The mapping function f_o	
function	event
Off	TurnOff
On	TurnOn
Idle	ToIdle
AAT	ToAAT
ChgLoc	PMLocCh

Figure 1: A DDM for a cardiac Pacemaker.



The mapping function f_o	
function	event
Off	SwitchOff
On	SwitchOn
Start	Start
Stop	Stop

Figure 2: A DDM for a Magnetic Resonance Imaging device.

$f_o : O \rightarrow \Sigma$ must be defined by all devices i.e. f_o maps the functionality of the device to an event of the finite state machine.

- $LS = \{l_0, l_1, \ldots, l_r\}$ is a set of labels where each $l_i, i = 1 \ldots r$ identifies a particular geographic location.
- $O = \{o_1, o_2, ..., o_u\}$ is the set of functions provided by the device. Invocation of $o_i, i = 1 \ldots u$ generates an event $\sigma \in \Sigma$.
- $A = \{a_1, a_2, \ldots, a_m$, *curState*, *curLoc*$\}$ is the set of attributes. $a_i, i = 1 \ldots m$ are the attributes exported by the device. Each a_i is of some type *type*(a_i). A contains two special attributes, *curState* $\in Q$ and *curLoc* $\in LS$, that must be exported by all devices. The current state and location of the device are obtained through *curState* and *curLoc* attributes, respectively. The *type(curState)* is Q, while the *type(curLoc)* is LS. The initial values of *curState* and *curLoc* are set to q_0 and l_0, respectively.

Figure 1 shows the DDM for a modified version of the cardiac Pacemaker described in [4]. The associated table in Figure 1 shows the mapping between the functions provided by the device and the events that trigger state transitions in the finite state machine. For example, when a user requests the Off function, the TurnOff event is generated. The event may cause a state transition in the finite state machine, thus changing the state of the device. Figure 2 shows the DDM for an Magnetic Resonance Imaging (MRI) device. The interference of an MRI device with a Pacemaker is well known [16]. It is possible to find the coexistence of these devices, for example, in an Imaging Room, in hospitals. These devices may be described using their DDM, and the

Imaging Room itself may be modeled as a ConnectedSpace. Subsequent sections in this paper show how the safety of the Imaging Room may be enforced using the safety policy listed in [16].

3 Safety Policies

The safe behavior of a ConnectedSpace is specified using a set of one or more safety policies. These safety policies may already be available for different environments such as health care and hospitals [16]. The safety policies for devices that may be found in and around a home are also available [1]. Various analysis techniques such as those based on fault-tree, event-tree, and petri nets [7, 8] may be used to derive these policies. The safety policies, when implemented correctly, govern the behavior of the ConnectedSpace with respect to safety.

A safety policy P is a 3-tuple given by $P=(I,O,r)$ where I is a boolean expression in the attributes and states of the devices in the ConnectedSpace, O is an ordered list of labels that identify a device, and $r \in \mathbb{N}$. The list O consists of the labels C_i for those devices whose states occur in I. A label C_i is associated with device D_i. O is represented as $<C_1, C_2, \ldots, C_k>$ where $\text{SCR}(P, D_i) < \text{SCR}(P, D_{i+1})$, $i = 1 \ldots k-1$. SCR: $P \times CS \rightarrow \mathbb{N}$ defines the rank of a device w.r.t a policy P. $\text{SCR}(P, D_i)$ is the *safety criticality rank* of D_i w.r.t P. The boolean expression I is a logical negation of a conjunctive clause formed using the Q_u states of one or more devices. The value of I must always be *true* i.e. I is an invariant.

A safety policy P is *violated* when I remains *false* for a duration greater than r. r is the *relaxation duration* for P. r may be assumed to be in appropriate time units. As mentioned earlier, it is possible that a ConnectedSpace is deployed in environments that differ widely on the desired safety criticality. These environments may have varying safety requirements. For example, it is possible that a laboratory may require a low safety criticality for certain policies i.e. they may tolerate violations of these policies for a brief period. Whereas, a health care unit may not tolerate any safety violation. Hence, it must be possible to specify and enforce such a behavior. Also, the same type of a device might have varying safety criticality. The device's safety criticality might depend on the environment. It may also depend on the individual policies that govern the device's behavior. Hence, it is necessary to provide the ability to rank the various devices with respect to their safety criticality.

The safety criticality rank provides

- a means by which the safety criticality of the ConnectedSpace can be tuned by adjusting the criticality of individual devices.
- a guide by which a less conservative controller may be synthesized i.e. when a transition, by an higher ranked device could potentially lead to a policy being violated, the controller, instead of disabling the transition, may move one or more of the lower ranked devices to new states in order to enable the current transition.

Consider the Imaging Room example described in Section 2. For such a ConnectedSpace, the safety policy described in [16] may be specified as PMSAFE = $\neg(curState_{PM} = \text{PACING} \wedge curState_{MRI} = \text{IMAGING})$ <PM,MRI>(25). The policy is interpreted as follows. The policy is violated when PM and the MRI device are in PACING and IMAGING states, respectively. Also, PM has a higher safety criticality rank than MRI. Hence, when the PM is in PACING, MRI must not be allowed to move to IMAGING. If MRI is already in IMAGING and if there is a request to move PM to PACING, then the safety controller has the option of temporarily moving MRI to a different state in order to permit the request. The safety controller may delay this move by no more than 25 time units.

4 Expressing ConnectedSpaces Using TTM

A ConnectedSpace may be expressed using a TTM [11, 12]. TTMs are based on state machines. The transitions of the state machine are augmented with data and control variables. A guard condition may also be specified for each transition. The delays and timeouts for each transition are modeled using lower and upper

Table 1: A timed transition model of PM.

τ	e_τ	h_τ	l_τ	u_τ
TurnOff	($curState_{PM}$=IDLE ∨ $curState_{PM}$=REFRACT ∨ $curState_{PM}$=PACING ∨ $curState_{PM}$=WAITING)	[$curState_{PM}$:OFF]	0	t_1
TurnOn	($curState_{PM}$=OFF)	[$curState_{PM}$:IDLE]	0	t_2
ToAAT	($curState_{PM}$=IDLE)	[$curState_{PM}$:REFRACT]	0	t_3
ToIdle	($curState_{PM}$=REFRACT)	[$curState_{PM}$:IDLE]	0	t_4
Wait	($curState_{PM}$=REFRACT)	[$curState_{PM}$:WAITING]	0	t_5
ASense	($curState_{PM}$=WAITING)	[$curState_{PM}$:PACING]	0	t_6
ASenseTM	($curState_{PM}$=WAITING)	[$curState_{PM}$:PACING]	0	t_7
PulseWidth	($curState_{PM}$=PACING)	[$curState_{PM}$:REFRACT]	0	t_8
PMLocCh	($curState_{PM}$=IDLE ∨ $curState_{PM}$=REFRACT ∨ $curState_{PM}$=PACING ∨ $curState_{PM}$=WAITING ∨ $curState_{PM}$=OFF)	—	0	t_9

time bounds, respectively. A TTM M is given by a 3-tuple $M = (\mathcal{V}, \Theta, \mathcal{T})$ where $\mathcal{V}$ is the set of variables that define M, Θ is a boolean-valued expression over the variables in $\mathcal{V}$ that defines the initial values of the variables, and $\mathcal{T}$ is the set of all transition of M. A transition $\tau \in \mathcal{T}$ is given by $(e_\tau, h_\tau, l_\tau, u_\tau)$ where e_τ is the enabling condition for τ, h_τ is the transformation function for the variables in $\mathcal{V}$, l_τ is the delay for τ, and h_τ is the deadline for τ. In addition to the variables on M, $\mathcal{V}$ also contains a special variable n. n gives the next transition for M.

Let $CS = \{D_1, D_2, \ldots, D_n\}$ be a ConnectedSpace where $D_i = (CSM_i, F_i, LS_i, O_i, A_i)$ and $CSM_i = (Q_i, \Sigma_i, \delta_i, \Omega_i, q_{0,i}, SD_i)$. The procedure to express the ConnectedSpace as a TTM is as follows.

- Each $D_i \in CS$ is expressed as a TTM $M_i = (\mathcal{V}_i, \Theta_i, \mathcal{T}_i)$.
- $\mathcal{V}_i = \{a_{k,i} | a_{k,i} \in A_i\} \cup \{curState_i \in A_i\} \cup \{curLoc_i \in A_i\}$ where $a_{k,i}$ is attribute a_k of device D_i.
- $\Theta_i = (n = initial,\ curState_i = q_{0,i} \in Q_i,\ curLoc_i = l_{0,i},\ a_{1,i} = a_{10,i}, a_{2,i} = a_{20,i}, \ldots, a_{k,i} = a_{k0,i})$ where $a_{k0,i}$ is the initial value of $a_{k,i}$, and $q_{0,i}$ is the initial state q_0 of device D_i.
- $\mathcal{T} = \{\tau | \tau \in \Sigma_i\}$ where $e_\tau = \vee_{j=1}^{k} (curState_i = q)\ \forall q \in Q_i$, and $\delta_i(q, \tau)!$, $h_\tau : curState_i = \delta_i(q, \tau)$, $l_\tau = 0$, and $u_\tau = \Omega_i(\tau)$.

The ConnectedSpace, which is the plant to be controlled, is then $PLANT$=($\mathcal{V}_{PLANT}$, Θ_{PLANT}, $\mathcal{T}_{PLANT}$) = ($\mathcal{V}_1 \cup \mathcal{V}_2 \cup \ldots \cup \mathcal{V}_n, \Theta_1 \wedge \Theta_2 \wedge \ldots \wedge \Theta_n, \mathcal{T}_1 \cup \mathcal{T}_2 \cup \ldots \cup \mathcal{T}_n$). It is assumed that there are no shared transitions among Σ_is. Hence, $\mathcal{T}_{PLANT}$ is computed as the union of $\mathcal{T}_i$s. Using the procedure listed, the DDM for PM described in Section 2 may be expressed as follows. $M_{PM} = (\mathcal{V}_{PM}, \Theta_{PM}, \mathcal{T}_{PM})$ where $\mathcal{V}_{PM} = (n, curState_{PM}, curLoc_{PM})$, $\Theta_{PM} = (n = initial, curState_{PM} = OFF, curLoc_{PM} = l_0)$. $\mathcal{T}_{PM}$ is shown in Table 1.

5 Expressing the Safety Policies using RTTL

Expressing the safety policies defined in Section 3 in RTTL is straightforward. This can be done by expressing each policy $P_k = (I_k, O_k, r_k)$ as a control specification $S_k \stackrel{def}{=} \Box I_k$. Thus, PMSAFE can now be specified as $S \stackrel{def}{=} \Box\neg(curState_{PM}$=PACING $\wedge curState_{MRI}$=IMAGING). S_k is interpreted as "henceforth I_k is *true*". I_k is the unsafe state defined by S_k. O_k and r_k are not specified in the control specification, but, instead,

are incorporated in the actions taken by the controller. The CDP is appropriately modified to synthesize the controller that enforces O_k and τ_k for each P_k.

6 Synthesis of the Safety Controller

Given the model of the plant as a TTM, and the control specifications in RTTL, a safety controller may be synthesized using CDP [11]. Based on the specifications, CDP identifies the unsafe states of the plant. The notions of partial weakest preconditions and partial strongest postconditions, are used to identify the transitions from a safe state to an unsafe state. Such transitions are termed *unsafe*. To avoid an exhaustive search in identifying the unsafe transitions, CDP works backwards starting from the unsafe state. The unsafe transitions are then made safe by disabling the transition or by forcing a safe transition. A controller synthesized using CDP is guaranteed to enforce the specifications.

Given a specification $S_k \stackrel{def}{=} \Box I_k$, for each transition $\tau \in \mathcal{T}_{PLANT}$, the safe state, $Prebad_{\tau,PLANT}$, from which the unsafe state I_k may be reached in a *single* step is computed. $Prebad_{\tau,PLANT}$ is defined as $Prebad_{\tau,PLANT} = pwp(\tau, \psi_k)$. $pwp(\tau, \psi_k)$ is the partial weakest precondition, and is defined as $pwp(\tau, \psi_k) = e_\tau \wedge \psi_k$ where ψ_k=I_k. Hence, τ is unsafe if $Prebad_{\tau,PLANT}$ exists i.e if $Prebad_{\tau,PLANT}$ is not *false*. Table 2 and Table 3 show $Prebad_{\tau,MRI}$ and $Prebad_{\tau,PM}$ states for MRI and PM, respectively. As there are no shared transitions among PM and MRI, the $Prebad_{\tau,PLANT} = Prebad_{\tau,MRI} \cup Prebad_{\tau,PM}$. The procedure described will result in two $Prebad_{\tau,PLANT}$ states: (a) ($curState_{PM}$ = PACING $\wedge$ $curState_{MRI}$ = MODESEL), and (b) ($curState_{PM}$ = WAITING $\wedge$ $curState_{MRI}$ = IMAGING).

After computing $Prebad_{\tau,PLANT}$, CDP may then be used to synthesize the controller. CDP consists of four different steps. Step 3 specifies two different strategies to handle a *pending* (unsafe) transition τ:

- R1: Change the enabling condition e_τ so that $Prebad_{\tau,PLANT}$ is *false*.
- R2: Force the occurrence of a safe transition σ *prior* to the occurrence of τ. This strategy is applicable only if $u_\sigma > l_\tau$. For ConnectedSpaces expressed using the procedure outlined earlier, $u_\sigma > l_\tau, \forall\sigma\forall\tau \in \mathcal{T}_{PLANT}$. Hence, it is always possible to apply R2.

Step 3 is modified, to incorporate the safety criticality rank and the relaxation duration. The modification to Step 3 ensures that each unsafe transition is made safe in a single step of the procedure. Hence, Step 4 of CDP is eliminated. The CDP modified for synthesizing safety controllers for ConnectedSpaces (CS-CDP) is shown in Figure 3.

Step 3 of CS-CDP ensures that a transition τ belonging to a higher ranked device is *always* enabled. In order to ensure that τ is enabled, the plant is moved to a safe state. During the process of moving to a safe state, the relaxation duration r_i of the current specification S_i is used to determine if τ can be forced first. The effect of the modification may be illustrated using PMSAFE. Let Stop be the safe transition, and let the maximum transition duration, t_{Stop}, be 20. In this case, if MRI is currently in state IMAGING, and ASense is requested, the controller will force ASense followed by Stop. When PM is in state PACING, and the transition Start is requested, it will be denied (disabled). Hence, a higher ranked device, such as PM in the example, will be allowed to function without hindrance while adhering to specifications.

The control actions computed using CS-CDP are used by the safety controller (SC) in the $Prebad_{\tau,PLANT}$ states. SC is also modeled as a finite state automaton. SC is given by a (a) 4-tuple $(Q_{SC}, \Sigma_{SC}, \delta_{SC}, q_0)$ where Q_{SC} is the set of states, Σ_{SC} is the set of events, δ_{SC} is the transition function, and q_0 is the initial state of SC, and (b) control table $CT : Q_{SC} \times \Sigma_{SC} \rightarrow$ control action for SC. A control action may disable, enable or a force one or more events. $CT(q)$, $q \in Q_{SC}$, defines the action that must be taken by SC on receiving an event $\tau \in \Sigma_{SC}$ when SC is in state q. SC is obtained as a *shuffle* product [15] i.e. $SC = M_1||M_2||\ldots||M_n$, $i = 1 \ldots n$. The $Prebad_{\tau,PLANT}$ states in SC are then augmented with the control actions. The safety controller SC for the Imaging Room example, without the self-loops due to MRILocCh and PMLocCh, would have 14 states and 66 transitions. Each state $q_i \in Q_{SC}$, $i = 0, \ldots, 13$ is given by $q_i = (q'_j, q''_k)$ where q'_j and q''_k are states of MRI and PM, respectively. The states $q_i \in Q_{SC}$, $i = 0, \ldots, 13$ are shown in Table 5. The states

Table 2: Computation of the $Prebad_{\tau,PM}$ states.

τ	e_τ	h_τ	$Prebad_{\tau,PM} = pwp(\tau, I_k)$
TurnOff	($curState_{PM}$ =IDLE ∨ $curState_{PM}$=REFRACT ∨ $curState_{PM}$=PACING ∨ $curState_{PM}$=WAITING)	[$curState_{PM}$:OFF]	$false$
TurnOn	($curState_{PM}$=OFF)	[$curState_{PM}$:IDLE]	$false$
ToAAT	($curState_{PM}$=IDLE)	[$curState_{PM}$:REFRACT]	$false$
Toldle	($curState_{PM}$=REFRACT)	[$curState_{PM}$:IDLE]	$false$
Wait	($curState_{PM}$=REFRACT)	[$curState_{PM}$:WAITING]	$false$
ASense	($curState_{PM}$=WAITING)	[$curState_{PM}$:PACING]	($curState_{PM}$=WAITING ∧$curState_{MRI}$=IMAGING)
ASenseTM	($curState_{PM}$=WAITING)	[$curState_{PM}$:PACING]	($curState_{PM}$=WAITING ∧$curState_{MRI}$ =IMAGING)
PulseWidth	($curState_{PM}$=PACING)	[$curState_{PM}$:REFRACT]	$false$
PMLocCh	($curState_{PM}$=IDLE ∨ $curState_{PM}$=REFRACT ∨ $curState_{PM}$=PACING ∨ $curState_{PM}$=WAITING ∨ $curState_{PM}$=OFF)	—	$false$

Table 3: Computation of the $Prebad_{\tau,MRI}$ states.

τ	e_τ	h_τ	$Prebad_{\tau,MRI} = pwp(\tau, I_k)$
SwitchOff	($curState_{MRI}$=MODESEL ∨ $curState_{MRI}$=IMAGING	[$curState_{MRI}$:OFF]	$false$
SwitchOn	($curState_{MRI}$=OFF)	[$curState_{MRI}$:MODESEL]	$false$
ToMode1	($curState_{MRI}$=MODESEL)	[$curState_{MRI}$:MODESEL]	$false$
ToMode2	($curState_{MRI}$=MODESEL)	[$curState_{MRI}$:MODESEL]	$false$
Start	($curState_{MRI}$=MODESEL)	[$curState_{MRI}$:IMAGING]	($curState_{PM}$=PACING $curState_{MRI}$=MODESEL)
Stop	($curState_{MRI}$=IMAGING)	[$curState_{MRI}$:MODESEL]	$false$
MRILocCh	($curState_{MRI}$=OFF ∨ $curState_{MRI}$=MODESEL ∨ $curState_{MRI}$=IMAGING)	—	$false$

PROCEDURE CS-CDP.

1. Check that Θ_{PLANT} is a safe state. If the check fails, then no additional control will achieve the required specification. If the check succeeds, then proceed to the Step 2.

2. If every transition is safe, then the plant already satisfies the control specification. Hence, exit from the procedure.

3. Mark every unsafe transition τ as *pending* i.e. it is yet to be made safe.

4. Make every pending transition safe as follows.
   if $SCR(P_k, D_j) > SCR(P_k, D_k)$, $\tau \in \Sigma_j$, $\sigma \in \Sigma_k$,
   if $(\tau_k \geq t_\sigma)$
   then force transition τ followed by σ. /* *Case 1: The ConnectedSpace is allowed to remain unsafe for a maximum duration of τ_k time units.* */
   else force transition σ followed by τ. /* *Case 2(a): The ConnectedSpace is not allowed to be become unsafe.* */
   else disable τ. /* *Case 2(b): Since the unsafe transition is requested on a device with a lower safety criticality rank, the ConnectedSpace is prevented from becoming unsafe by denying (disabling) the transition.* */

Figure 3: Procedure CS-CDP for synthesizing safety controllers for ConnectedSpaces.

Table 4: The control table for the safety controller.

$q \in Q_{SC}$	$\tau \in \Sigma_{SC}$	$\delta_{SC}(q, e)$	$\sigma \in \Sigma_{SC}$	controller action
q_{12}	ASense	q_{13}	Stop	enable τ and then force σ
q_{12}	ASenseTM	q_{13}	Stop	enable τ and then force σ
q_{13}	Start	q_{13}	-	disable τ

q_{12} = (IMAGING, WAITING) and q_{13} = (MODESEL, PACING) in SC correspond to the $Prebad_{\tau,PLANT}$ states. The control table and δ_{SC} are shown in Table 4, and Table 6, respectively. For brevity, the control table only lists those states of SC where an event is either disabled or forced. When the controller is in the states that are not specified in CT, it enables all events. Also, note that Case 1, Step 3 of CS-CDP ensures that an unsafe state of the ConnectedSpace, and hence the controller, is transitory. Hence the (IMAGING, PACING) state of the controller is not listed in Table 5. For example, when the controller is in state q_{12} = (IMAGING, WAITING) and ASense transition is requested, the controller moves to state (MODESEL, PACING) through state (IMAGING, PACING). Hence, $\delta_{SC}(q_{12}, ASense)$ will be q_{13}.

7 Experimental Evaluation

CS-CDP and the procedure for computing the $Prebad_{\tau,PLANT}$ states were implemented in C++ using the LEDA [9] libraries. The implementation was used to experiment with CS-CDP. The experiments were conducted on a Pentium-III machine, with 256MB of RAM, operating at 700MHz, and running the Linux operating system.

Consider a ConnectedSpace CS=$\{D_1, D_2, \ldots, D_m\}$ where $D_i, i = 1 \ldots m$ is the DDM for device D_i. Let SP=$\{P1, P2, \ldots P_n\}$ be the set of safety policies imposed on CS. The *size* of a ConnectedSpace is specified by the number of policies $n = |SP|$, and the number of devices $m = |CS|$. ConnectedSpaces are classified, based on their size, as (a) `small`, (b) `medium`, (c) `large`, and (d) huge. The set of safety policies is characterized

Table 5: The states of the safety controller.

$q_i \in Q_{SC}$	(q'_j, q''_k)
q_0	(OFF, OFF)
q_1	(MODESEL, OFF)
q_2	(OFF, IDLE)
q_3	(IMAGING, OFF)
q_4	(MODESEL, IDLE)
q_5	(OFF, REFRACT)
q_6	(IMAGING, IDLE)
q_7	(MODESEL, REFRACT)
q_8	(OFF, WAITING)
q_9	(IMAGING, REFRACT)
q_{10}	(MODESEL, WAITING)
q_{11}	(OFF, PACING)
q_{12}	(IMAGING, WAITING)
q_{13}	(MODESEL, PACING)

using the notion of `density`. The density d of SP is defined as $d = \frac{\sum_{i=1}^{n} C(P_i)}{n*m}$ where C : SP$\rightarrow$ $\mathbb{N}$ is a function that gives the number of unique devices whose states or attributes occur in a policy.

The average number of states s in the finite state machines is defined as $s = \frac{\sum_{i=1}^{n} |Q_i|}{m}$ where Q_i is the set of states for D_i. The CPU usage, in milliseconds, for computing the $Prebad_{\tau,PLANT}$ states and for executing CS-CDP, are denoted by $T_{\texttt{Prebad}}$ and $T_{\texttt{CS-CDP}}$, respectively. The total CPU usage $T_{\texttt{total}}$ for synthesizing the safety controller is the sum of $T_{\texttt{Prebad}}$ and $T_{\texttt{CS-CDP}}$.

For each of the four classes of ConnectedSpaces, experiments were conducted to study the following:

- The impact of n and m on $T_{\texttt{total}}$.
- The impact of d on $T_{\texttt{total}}$.
- The impact of s on $T_{\texttt{total}}$.

For each class of ConnectedSpace, representative values for n, m, d, and s are chosen. These values are used to generate SP uniformly at random. The extended finite state machine CSM_i for device $D_i \in CS$, $i = 1 \ldots m$ is generated as follows.

- The set of states $Q_i = \{Q_s \cup Q_u\} \in CSM_i$ is generated such that $|Q_i| = s$ and $|Q_s| = 1$.
- The set of events $\Sigma_i \in CSM_i$ is generated such that $|\Sigma_i| = n(n-1)$.
- The state transition function $\delta_i \in CSM_i$ is generated such that (a) there is a transition from each state in Q_u to every state in Q_s, (b) there is a transition from each state in Q_s to every state in Q_u, and (c) there is a transition from each state in $q \in Q_u$ to every other state in Q_u. Each $\tau \in \Sigma$ triggers exactly one transition.
- The cost function $\Omega_i \in CSM_i$ is generated such that $\Omega(\tau) = 500$, $\delta(q, \tau) \in Q_s$, $q \in Q_u$ i.e. a maximum duration of 500 time units is assigned to every transition from a state in Q_u to a state in Q_s. For all other transitions, a duration in $[1, 80]$ time units is assigned uniformly at random.
- The initial state $q_{0,i}$ of D_i is the state in Q_s.

Table 6: The transition function for the safety controller.

$q \in Q_{SC}$	$e \in \Sigma_{SC}$	$\delta_{SC}(q,e)$
q_0	SwitchOn	q_1
q_0	TurnOn	q_2
q_1	SwitchOff	q_3
q_1	Start	q_0
q_1	ToMode1	q_1
q_1	ToMode2	q_1
q_1	TurnOn	q_4
q_2	SwitchOn	q_4
q_2	TurnOff	q_0
q_2	ToAAT	q_5
q_3	Stop	q_1
q_3	SwitchOff	q_0
q_3	TurnOn	q_6
q_4	SwitchOff	q_2
q_4	TurnOn	q_6
q_4	ToMode1	q_4
q_4	ToMode2	q_4
q_4	TurnOff	q_1
q_4	ToAAT	q_7
q_5	SwitchOn	q_7
q_5	TurnOff	q_0
q_5	ToIdle	q_2
q_5	Waiting	q_8
q_6	Stop	q_4
q_6	SwitchOff	q_2
q_6	TurnOff	q_3
q_6	ToAAT	q_9
q_7	SwitchOff	q_5
q_7	Start	q_9
q_7	ToMode1	q_7
q_7	ToMode2	q_7
q_7	TurnOff	q_1
q_7	ToIdle	q_4

$q \in Q_{SC}$	$e \in \Sigma_{SC}$	$\delta_{SC}(q,e)$
q_7	Waiting	q_{10}
q_8	SwitchOn	q_{10}
q_8	TurnOff	q_0
q_8	ToIdle	q_2
q_8	ASense	q_{11}
q_9	ASenseTM	q_{12}
q_9	Stop	q_7
q_9	SwitchOff	q_5
q_9	TurnOff	q_3
q_9	ToIdle	q_6
q_9	Waiting	q_{12}
q_{10}	SwitchOff	q_8
q_{10}	Start	q_{12}
q_{10}	ToMode1	q_{10}
q_{10}	ToMode2	q_{10}
q_{10}	TurnOff	q_1
q_{10}	ToIdle	q_4
q_{10}	ASense	q_{13}
q_{10}	ASenseTM	q_{13}
q_{11}	SwitchOff	q_{10}
q_{11}	TurnOff	q_0
q_{11}	ToIdle	q_2
q_{11}	PulseWidth	q_5
q_{12}	ASense	q_{13}
q_{12}	ASenseTM	q_{13}
q_{12}	Stop	q_{10}
q_{12}	SwitchOff	q_8
q_{12}	TurnOff	q_3
q_{12}	ToIdle	q_6
q_{13}	Start	q_{13}
q_{13}	SwitchOff	q_{11}
q_{13}	ToMode1	q_{13}
q_{13}	ToMode2	q_{17}
q_{13}	ToIdle	q_4
q_{13}	PulseWidth	q_5

A single run of the experiment consists of executing CS-CDP for (a) $d = 0.1, 0.2, \ldots, 1$, and (b) $s = 5, 10, 15, \ldots, 40$, for each class of ConnectedSpace. Values of $T_{\texttt{Prebad}}$ and $T_{\texttt{CS-CDP}}$ are collected for each run. The value of $T_{\texttt{total}}$ is then computed using the average values of $T_{\texttt{Prebad}}$ and $T_{\texttt{CS-CDP}}$ over 10 runs of the experiment.

Table 7 shows the representative values of n, m, s, and $T_{\texttt{total}}$ for $d = 0.5$, for each of the four classes. As may be observed from Table 7, the total CPU usage increases linearly with increase in the size of the ConnectedSpace.

The impact of the number of devices and the number of policies on the $T_{\texttt{total}}$ are shown in Figure 4 and Figure 5, respectively. As may be observed from the figures, the total CPU usage increases linearly with increasing number of devices and policies.

The impact of d and s on $T_{\texttt{total}}$ are shown in Figure 6 and Figure 7, respectively. The figures show that $T_{\texttt{total}}$ increases linearly with respect to both d and s.

Table 7: The characteristic values of n, m, s, and $T_{\texttt{total}}$ for each of the four classes of ConnectedSpaces for $d = 0.5$.

	No. of policies n	No. of devices m	Average no. of states s	CPU usage for synthesizing the safety controller (milliseconds) $T_{\texttt{total}}$
small	5	20	20	0.541
medium	25	100	20	19.139
large	125	700	20	726.65
huge	250	2000	20	3949.54

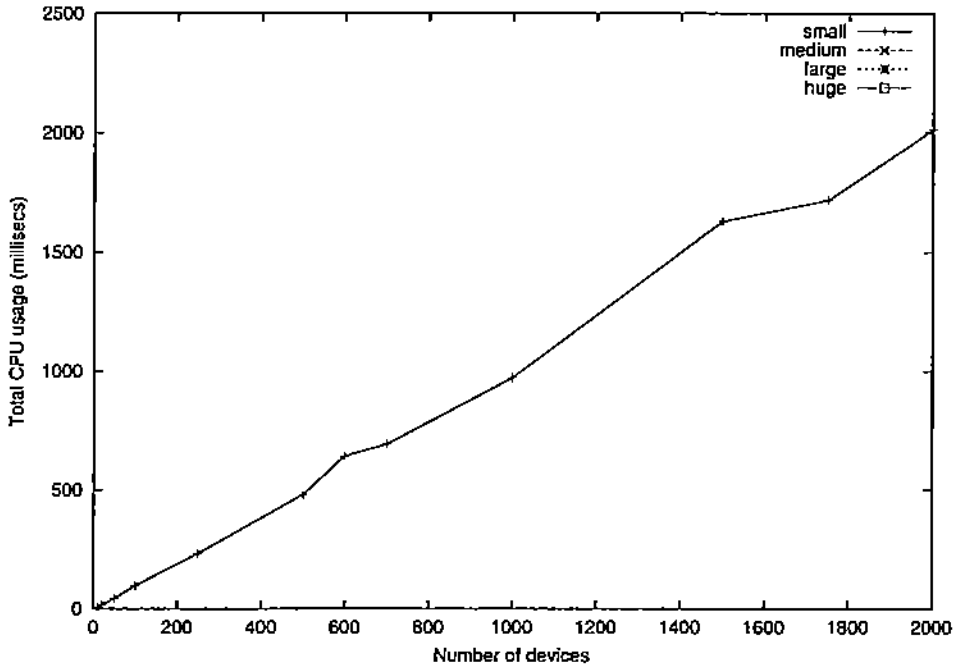


Figure 4: The effect of the number of devices m on the total CPU usage $T_{\texttt{total}}$ for $n = 125$.

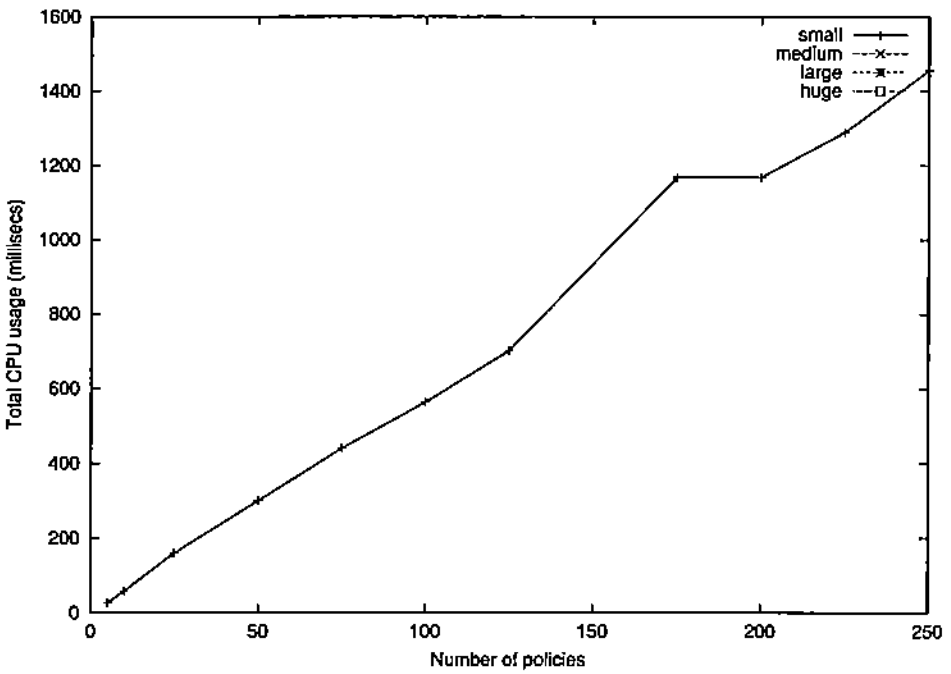


Figure 5: The effect of the number of policies n on the total CPU usage $T_{\texttt{total}}$ for $m = 700$.

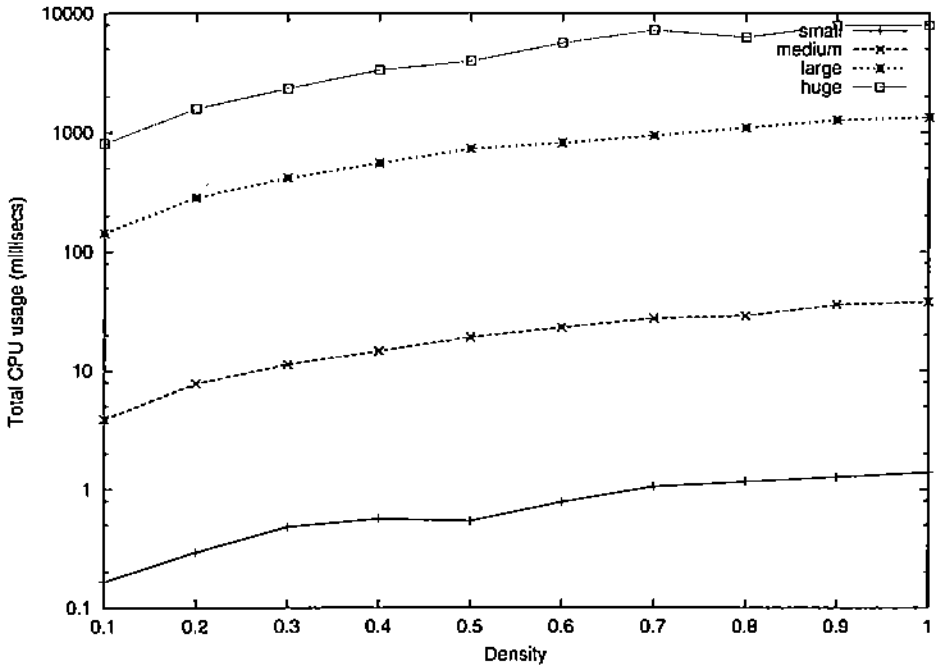


Figure 6: The effect of density d on the total CPU usage $T_{\texttt{total}}$ for $s = 20$.

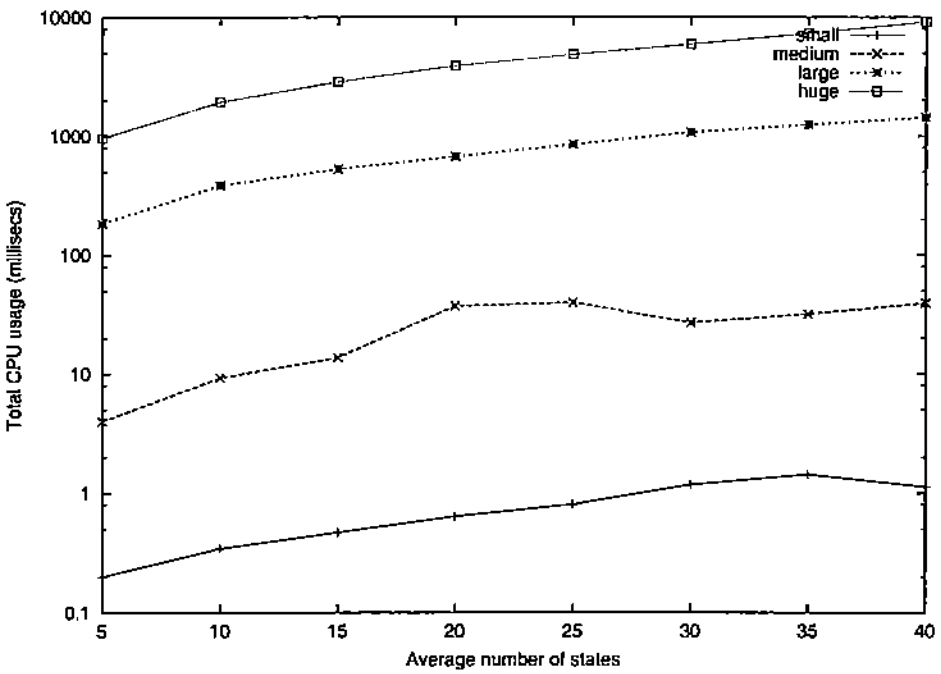


Figure 7: The variation of the total CPU usage $T_{\texttt{total}}$ with respect to the average number of states of the finite state machines for $d = 0.5$.

8 Future Work

ConnectedSpaces and Digital Device Manuals are described in the paper. The notions of safety criticality rank and policy relaxation duration were introduced. The approach for applying the TTM/RTTL framework in the synthesis of a safety controller for ConnectedSpaces is described. A sample ConnectedSpace in a hospital environment is used to illustrate the approach. An experimental evaluation of the synthesis procedure is also presented. The experiments show that the procedure is scalable with respect to the number of devices, number of policies, and the number of states of the finite state machines.

A ConnectedSpace may be highly dynamic in nature. For example, it is possible that the number and type of devices may change over time. In addition, the safety policies may also change. In such cases, the controller may have to be synthesized dynamically. Though the approach described in this paper is scalable, yet the time to compute the safety controller may not be acceptable in such ConnectedSpaces. Hence, an approach for the incremental synthesis of the safety controller may be needed. It may be possible to apply the modular synthesis approach presented in [13]. The issues involved in the development of an incremental approach for the controller synthesis need to be investigated.

In the control of critical properties such as safety, the fault-tolerance of the controller itself is crucial. The issues involved in the dynamic, automated synthesis of safety controllers, that are correct as well as fault-tolerant, need to be studied.

References

[1] Environmental Protection Agency. Carbon monoxide poisoning alert. *http://www.epa.gov/iaq/pubs/coalert.html.*

[2] B. A. Brandin and W. M. Wonham. Supervisory control of timed discrete-event systems. In *Proceedings of the 31st IEEE Conference on Decision and Control*, pages 3357–3362, Tucson, Arizona, USA, December 1992.

[3] Consumer Product Safety Commission. Safety for older consumers: Home safety checklist. *http://www.cpsc.gov/cpscpub/pubs/701.html.*

[4] B. P. Douglass. *Real-Time UML.* Addison Wesley Longman, Inc, 1998.

[5] Aviation Subcommitee Hearing. Portable electronic devices: Do they really pose a safety hazard on aircraft? http://www.house.gov/transportation/aviation/hearing/07-20-00/07-20-00memo.html.

[6] Pacemaker Related Incident. http://www.users.on.net/vision/misc/pacemaker-death.html.

[7] N. G. Leveson. *Safeware:System Safety and Computers.* Addison-Wesley Publishing Company, 1995.

[8] N. G. Leveson and J. L. Stolzy. Safety analysis using petri nets. *Transactions on Software Engineering*, 13(3):386–397, March 1987.

[9] K. Mehlhorn and S. N. Naher. LEDA, a library of efficient data types and algorithms. In *Proceedings of the 14th International Symposium on Mathematical Foundations of Computer Science*, volume 379 of *Lecture Notes in Computer Science*, pages 88–106. Springer-Verlag, 1989.

[10] NASA. Aviation safety reporting system. http://www.ire.org/datalibrary/databases/Asrs/.

[11] J. S. Ostroff. Synthesis of controllers for real-time discrete event systems. In *Proceedings of the IEEE 28th conference on Decision and Control*, pages 138–144, 1989.

[12] J. S. Ostroff. A logic for real-time discrete event processes. *IEEE Control Systems Magazine*, 10(4):95–102, June 1990.

[13] P. J. Ramadge and W. M. Wonham. Modular feedback logic for discrete event systems. *SIAM Journal of Control and Optimization*, 25(5):1202–1218, May 1987.

[14] P. J. Ramadge and W. M. Wonham. On a supremal controllable sublanguage of a given language. *SIAM Journal of Control and Optimization*, 25(3):637–659, May 1987.

[15] P. J. Ramadge and W. M. Wonham. Supervisory control of a class of discrete event processes. *SIAM Journal of Control and Optimization*, 25(1):206–230, January 1987.

[16] Medical Device Safety Report. Thoracic impedance measurements can interfere with impedance-based rate-responsive pacemakers. http://www.mdsr.ecri.org/.

[17] Medical Device Safety Reports. Audible Low-Battery alarms on pulseoximeters. *Health Devices*, 9(5-6):200, May-June 1990.

[18] Medical Device Safety Reports. Interference between Telemetry transmitters programmed to the same frequency. *Health Devices*, 28(1-2):82–83, Jan-Feb 1999.

[19] Home Safety. volume http://www.safewithin.com/homesafe/home.env.cgi.

[20] Pacemaker Safety. http://www.research-projects.unizh.ch/med/unit40600/area230/p2803.htm.

[21] B. Sridharan, A. P. Mathur, and S. Ungar. Digital Device Manuals for the management of ConnectedSpaces. *IEEE Communications Magazine*, 40(8):78–85, August 2002.