Quality of Service (QoS) in Healthcare Applications:

Colored Petri Net Simulation for Design of Heterogeneous, Multi-Vendor, Integrated, Life-Critical Wireless (802.x) Patient Care Device Networks

Vijay Gehlot
Villanova University
vijay.gehlot@villanova.edu

Elliot B. Sloane
Villanova University
ebsloane@ieee.org

ABSTRACT

The ability to deploy wireless patient monitors using industry-standard IEEE 802.x technologies allows patient mobility and clinical flexibly. However, interconnecting multiple life-critical medical devices from multiple vendors can introduce unintended life-threatening risks unless delivery of critical patient alarms to central monitoring systems and/or clinical personnel is assured. Petri net tools allow automated testing of all possible states and transitions between devices and/or systems to detect potential failure modes in advance. Colored Petri Net (CPN) tools allow tracking and controlling each message in a network based on pre-selected criteria. This paper describes a research project using CPN to simulate and validate alarm integrity in a small multi-modality wireless patient monitoring system. Free research CPN software, CPNTool, is used to simulate two, 20-monitor wireless patient monitoring networks. One network simulated standard non-prioritized 802.x IP protocols and simulated Quality of Service (QoS) capabilities similar to 802.11e, allowing message priority management. In the standard 802.x network, dangerous heart arrhythmia and pulse oximetry alarms were missed, but QoS priority management reduced that risk significantly.

Keywords

Colored Petri Nets, software validation, system validation, medical devices, medical information systems, wireless network security, Wi-Fi, IEEE 802.11e, patient safety, medical errors, HIPAA

INTRODUCTION

Advances in IT and related fields are rapidly changing the way health care is managed and delivered. Some manufacturers and hospitals are leveraging the integration of medical device and information system technologies to provide wireless LAN-based medical device architectures (Briggs 2004, Cohen 2001, Health Consultants 2003, Sloane 2001, Sloane 2002). On the one hand, such advances have been a boon, as they allow patients more healthy mobility and allow the hospital to provide clinical care in more flexible locations throughout the facility. Many of the emerging systems take advantage of cheap, widely available Wi-Fi (IEEE 802.11x) or Bluetooth (IEEE 802.15x) wireless network software and hardware that are based on industry-standard network protocols. The most common Wi-Fi versions (802.11a, 802.11b, 802.11g, and the emerging 802.11e) all use Internet Protocol (IP) network designs that are based on random collision detection (CD) algorithms to share the limited available time and frequency capacity with all users.

Digital messages from medical devices, like alarms or clinical data streams, are broken into small packets and sent through the device's antenna to a shared wireless network access point (typically on the ceiling.) For patient data, like heart monitors, each device may be constantly sending a stream of packets that can be assembled in correct sequence at a central station or computer system. When more than one packet of data arrives at the same time from different devices, the CD protocol senses the collision and each device is designed to wait a random delay period before resending the packet. Thus, the larger the number of medical devices that are sharing the access point, and the more data they are trying to constantly send, the more CD events there will be. In this type of standard wireless network, all data packets are treated as if they are "alike." Because most physiological waveforms have relatively low data rates and are not life-critical, these delays may not be visible or important.

Life-critical alarms, which may signal a heart attack, for example, cannot afford significant delays, however. Furthermore as the hospital deploys more and more wireless monitoring systems, multiple devices, representing multiple potential life-critical monitoring modalities, may try to send an unpredictable quantity of vendor-specific alarm packets through the network. This, then, is the crux of the situation that this study investigated. Since there is only a random chance of critical alarms getting through a wireless network when lots of data is being sent by multiple medical devices, we set up two

Abstract Published in Proceedings of the Eleventh Americas Conference on Information Systems, Omaha, NE, USA August 11th-14th 2005

simulation experiments. In the first, a simple random CD-type network was simulated. In the second simulation, we implemented a network priority management scheme that is often referred to as Quality of Service (QoS). The use of QoS is widely being discussed by vendors such as Cisco Systems in the Voice over Internet Protocol (VoIP) because is helps avoid unacceptable gaps and wavering in telephone-type discussions when using 802.x networks. QoS is covered in a supplemental IEEE standard known as 802.11e which must be applied in addition to the other basic 802.11x protocols. Unfortunately, 802.11e has not yet emerged in the form of industry-wide standard off-the-shelf compatible network components.

Multiple devices may constantly be sending streams of packets of non-time-critical data. Although CD allows it, but on the other hand this has meant that we are moving towards integration and automation of increasingly complex and critical processes and tasks with little or no human oversight or intervention. The risks of system failures or defects can be very serious; government reports have documented that as many as 90,000 American patients may be injured or killed by medical errors each year, and complex systems with defects could only make the problem worse (Institute of Medicine 1998). Although medical device manufacturers must validate their software-based products to FDA standards, no such requirements exist for hospitals.

The central problem with hybrid, multi-vendor, multi-generational devices and information systems is that the complexity of the interaction of the underlying systems multiplies with the introduction of each new subsystem. Safe and reliable operation of such systems depends heavily on their ability to function as desired, but systems of validation are not presently available in healthcare.

One way that has been used to ascertain desired behavior of a system in other industries is to make use of a formal modeling and validation/verification techniques. There are a good variety of such techniques that have been applied in a different context (Gehlot and Lee 1988). This current research is a first attempt to apply such techniques in health-care settings and to show that there are benefits to be drawn from such an approach. In particular, we have tried to show by means of a simple example how Petri Nets (Reisig 1985) may be used to model healthcare scenarios.

Petri nets are a good tool for modeling systems with interacting concurrent components. Their timed extensions have been used in performance modeling. The fundamental idea behind such modeling is that systems are composed of separate interacting components. Each component has its own state of being and its state may change over time via interactions. Furthermore, as a modeling tool, Petri nets have the following advantages:

- Flexibility: there are a wide range of extensions to suit different needs. For example, timed and stochastic extensions of Petri nets make them suitable for performance analysis.
- Adaptable: since they are based on very few key abstract ideas, they are easily adaptable to a variety of modeling domains.
- Simple syntax: no need to worry about various operators and their precedence etc.
- Simple semantics: there is just one rule, namely, the firing rule that describes the dynamic behavior.
- Visual: Petri nets are a graphical modeling notation. This makes them easy to understand and work with.
- Analysis tools: there are a good number of verification/validation tools available for Petri nets.

In the next section we give a formal definition of a Petri Net and show how a verification/validation tool can help detect design/interface problems. Following this we take a simple example from the medical/clinical domain and show how we can represent it using Petri nets and how we would approach verification/validation of the system prior to deployment. We present out conclusion and future work in the last section.

PETRI NETS

Petri nets are described in prior articles (Gehlot and Sloane 2004). In brief, the abstraction of a Petri net includes “state-like” objects (*S*) and “event-like” objects (*T*) and dependencies between these objects (*F*). The basic idea is that “any” phenomena or system can be described in terms of “cause and effect”. The state-like objects become the cause for the event-like objects to “occur” and the effect of which is ``another'' state-like object. Thus, A Petri net consists of the following:

- A finite set of *states* or *places* (denoted *S*)
- A finite set (disjoint from *S*) of *transitions* or *events* (denoted *T*)

- A finite subset of (S × T) ∪ (T × S) called the *flow relation* or the *dependency relation* (denoted *F*)
- A mapping from *S* to natural numbers (including infinity) called marking (denoted *M*), i.e., $M : S \rightarrow \mathcal{N}$

We have employed an extension of these Petri nets in our current approach. We give a very brief overview of this extended model next. A fuller description, including formal definitions, and details of dynamic behavior can be found in (Jensen 1997). We would like to note that there are many other approaches to formal modeling (Wing et al. 1999). Our justification for the use of Petri net based approach lies in our own familiarity with this approach and availability of free software tools.

COLORED PETRI NETS

Colored Petri Nets (CPNs) are an enhancement of Petri Nets in that the tokens have "colors" or types. This allows one to distinguish various types of tokens. This enhancement is attractive in our situation where we would like to analyze behaviors of different kinds of alarms. Furthermore, the CPN Tool, which is a computer tool for editing, modeling, simulating, and analyzing CPNs employs a powerful programming language called CPN ML for declarations and net inscriptions (Ratzer et al. 2003). This language is an extension of a well-known functional programming language called ML. This linguistic support within the tool allows one to model varieties of behaviors which are otherwise impossible to model in a Petri Net. In our case, we define several functions using CPN ML to impose some quality of service requirements. The CPN Tool also includes the timed extension of CPNs which is useful in capturing and simulating temporal behavior of a system. The CPN view of time-dependent events is different from the one in traditional Timed Petri nets (Gehlot 1988). The time concept in CPN is based on the notion of a global clock which represents model time (not actual physical time). A token may optionally be declared as a timed token. Such tokens carry a time stamp through simulation. The time stamp governs the availability of such tokens. We use timed tokens to govern the generation and handling of various alarms.

Colored Petri Nets Analysis

There are various approaches to analyzing a CPN. One approach is interactive/automatic simulation. The purpose of this approach is to investigate performance and identify bottlenecks. A second analysis method is based on state space or reachability graphs (also called an occurrence graph). Under this approach a directed graph whose nodes represent system state and whose edges represent the next state transitions is constructed (possibly automatically) and analyzed. It is well-known that the size of such graphs may be quite large, however, there exist techniques that allow one to work with a condensed graph. A third technique is based on net invariants and is useful in establishing certain structural properties. For our analysis, we employed the simulation approach since our main focus was system performance guarantees and identification of bottlenecks.

CPN MODEL OF WIRELESS MEDICAL DEVICE NETWORKS

We extended our basic model of a patient monitoring system presented in (Gehlot and Sloane 2004) and created a model using Colored Petri Nets. Figure 1 depicts this model. In creating this model, we separated three categories (colors) of alarms: Red, Orange, and Yellow. The Red alarms are generated by heart monitors (transition labeled **HMAlarm**). The Orange alarms are pulse oximetry alarms (transition labeled **POAlarm**) and the Yellow alarms indicate low heart monitor battery situations that could signal the impending complete failure of a heart alarm monitor (transition labeled **LoBatAlarm**). Furthermore, we used CPN's index color set facility to define and therefore track the heart alarms individually through the network,

To introduce time, specific functions were written to control enabling of transitions in order to simulate realistic alarm events. This involved introducing timed tokens in places that were attached to respective transitions. Essentially, these places (labeled D1, D2, and D3 in Figure 1) achieve a "gating" function. The Red heart alarms fired infrequently, as one would expect for recovering patients. The Yellow battery alarms occurred least frequently because good hospital maintenance practices should ensure stable battery performance. The Orange pulse oximetry alarms occurred quite frequently, as they are prone to many false-positives due to patient movement sensitivity. The timings were programmed to occur at intervals closely similar to the prior published values (Gehlot and Sloane 2004), and they are described with each simulation below.

In Figure 1, the oval shaped place labeled **802.11** represents a wireless LAN network access point. All alarms and reset request go through this network fabric. The packet queues are modeled using CPN ML's list data structure. The central station is modeled as two nurses managing the monitors and alarms (transition labeled **HandleAlarm**).

SIMULATION OF STANDARD 802.11X WIRELESS NETWORK AND THE QOS-ENHANCED NETWORK

We used the CPN Tool to run several simulations of the model described above. The parameters of simulations were set as follows:

- color OrangeDelay = int with 1..2;
- color YellowDelay = int with 100..500;
- color RedDelay = int with 2..50;
- color NurseDelayY= int with 1..5;

which represent random values uniformly distributed over the specified range. Since there are no quality of service guarantees associated with the underlying network fabric, we found situations in our simulation runs where a Red alarm may be significantly delayed. A snap-shot of one such scenario is depicted in Figure 1 above where we see the queue consisting of **[orange, orange, orange, orange, orange, R(red(1)), orange]**. This means that the heart monitor alarm from patient 1 may not get handled until the five pulse oximetry alarms get cleared. One idea to help alleviate such situations is to have some priority assignments and quality of service guarantees over the network. In addition, the handling of alarms should also be prioritized.

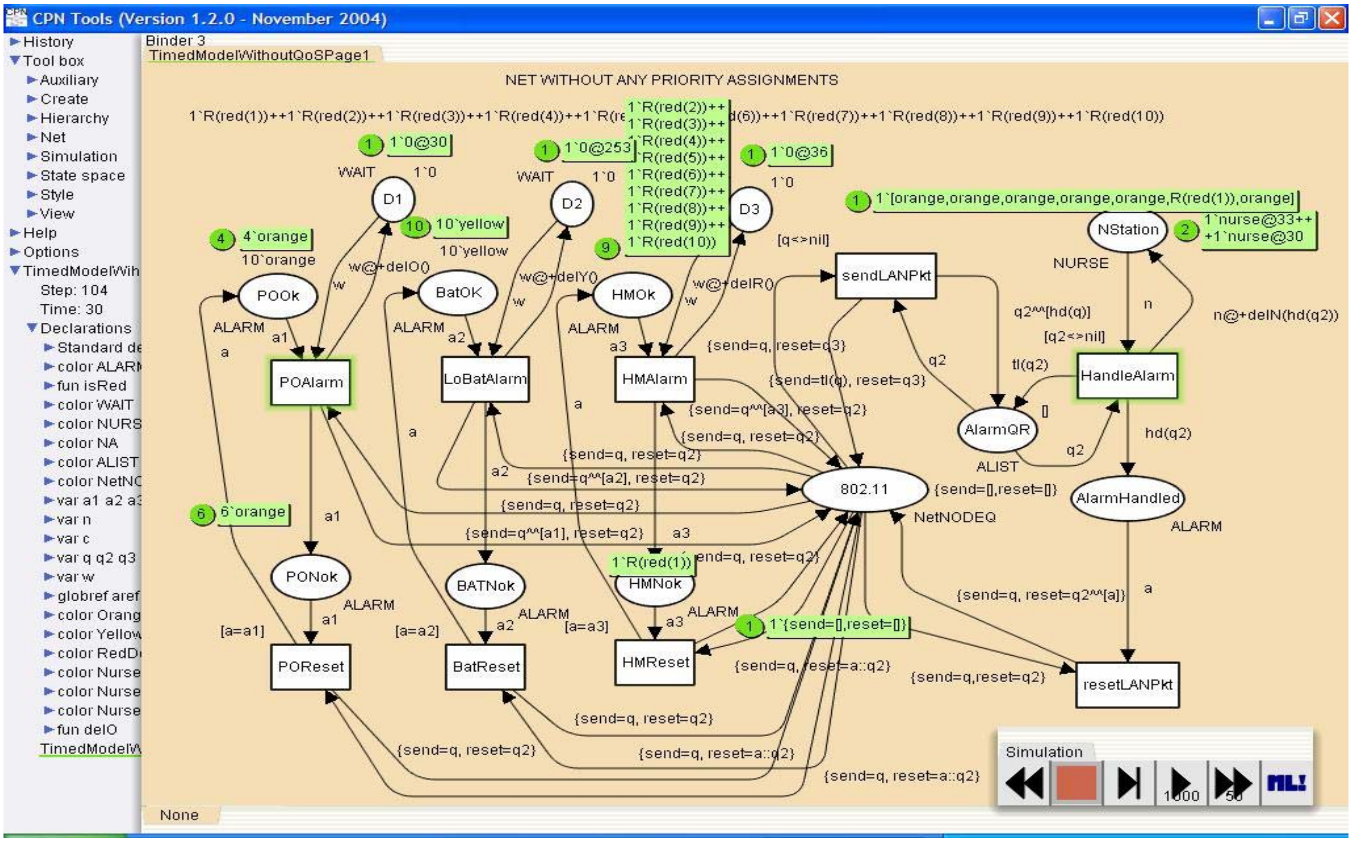

Figure 1. Colored Peti net model for medical device network without Quality of Service priority scheme.

We incorporated the QoS requirements in a modified version of the model depicted in Figure 1. The modification entailed a priority based queuing policy implemented as CPN ML functions below. In essence, these programs are designed to prioritize the passage of Red tokens ahead of Yellow ones, and Yellow ones ahead of Orange ones to ensure that critical heart alarms or heart monitor battery failure alarms propagate ahead of the less critical pulse oximetry alarms.

```
fun isRed(R(red(_)))= true
|     isRed(_) = false;

fun isYel(yellow) = true
|     isYel(_) = false;

fun hasRed(l) = List.exists isRed l;

fun hasYel(l) = List.exists isYel l;

fun remR (l) =
   if l=[] then []
   else if isRed(hd(l)) then tl(l)
 else hd(l) :: remR(tl(l));

fun getFstR(x::y) = if isRed(x) then x
   else getFstR(y);

fun remY (l) =
   if l=[] then []
   else if hd(l)=yellow then tl(l)
 else hd(l) :: remY(tl(l));

fun RYtl(l) =
   if hasRed(l) then remR(l)
   else if hasYel(l) then remY(l)
 else tl(l);

fun RYhd(l) =
   if hasRed(l) then getFstR(l)
   else if hasYel(l) then yellow
 else hd(l);
```

The main functions in these simple ML programs are **RYhd** and **RYtl.** These give priority to Red and Yellow alarm over Orange alarm. Furthermore, a Red alarm has priority over Yellow alarm. With these modifications incorporated, we ran several simulation runs and found no Red alarms getting delayed. The modified CPN, which is very similar to the one in Figure 1 above, has the network node labeled **802.11e** to indicate it has a form of QoS built-in.

Discrete Red tokens (red(1) through red(10)) can be seen in both Figure 1 and Figure 2. This allowed timing the delay of each of ten potential heart monitor alarms through the entire network. Neither Orange nor Yellow tokens were individually tracked in these simulations, as they are not immediately life-critical.

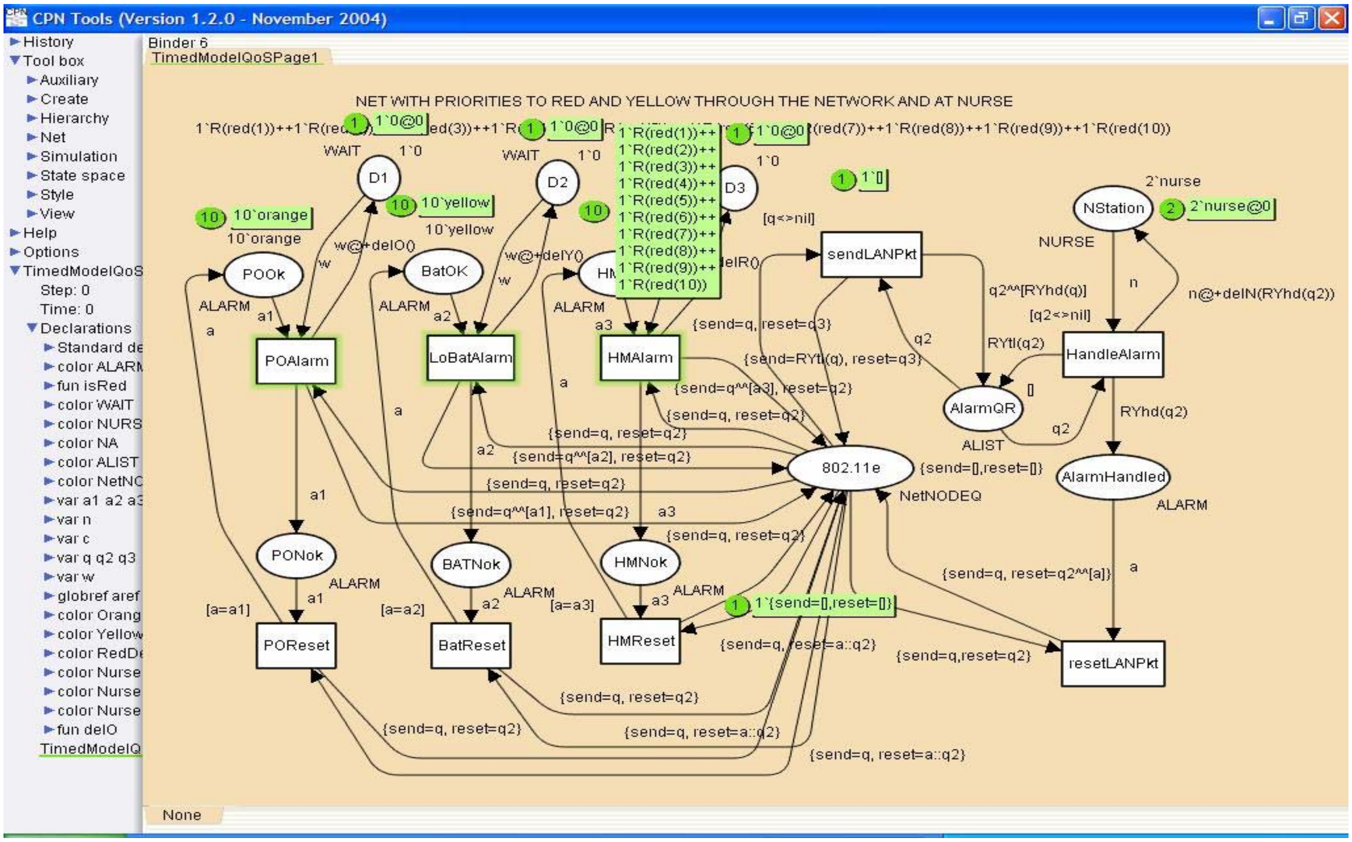

Figure 2. Colored Peti net model for medical device network with Quality of Service priority scheme for alarms.

RESULTS AND CONCLUSIONS

In the simulation shown in Figure 1 (representative of a standard 802.11x wireless network) we found that the heart alarms were delayed a minimum of 2 minutes, with a maximum delay 17 minutes. The average delay was over 10 minutes! In the QoS simulation shown in Figure 2, the minimum delay was 0, the maximum (rare) was 4, and the average delay was about 1.5.

The results from the simulation shown in Figure 1 obviously represent a totally unacceptable risk profile, as even a two-minute delay could cause death or irreparable brain damage. It is useful, perhaps, to note that the problem is not the total bandwidth of the system but the random nature of the collision detection mechanism. In our simulation, more frequent alarms from the pulse oximeters monitors simply cause more collisions, delaying all of the alarms in the network. This is by no means the most severe condition that a hospital could actually encounter, either. If, for example, the wireless network were shared with other, non-clinical uses, doctors with laptops or PDA's could tie the access points up with lengthy email or patient chart data (including images or multimedia content).

Although Figure 2's simulation was very much improved, the occasional 4-minute heart alarm delay is unacceptable. Further network design will be necessary in order to ensure that no heart alarm will be delayed more than about 30 seconds.

This pair of simple CPN simulations suggests several practical conclusions:

1. Design and deployment of a wireless patient monitoring network may not be a simple process

2. Use of common, industry standard 802.11x (Wi-Fi) network components may create an artificial sense of security, as their successful use for less risky network applications is not at all like life-critical medical signals;
3. QoS-compliant network equipment may be necessary for life-critical applications;
4. CPN tools can be programmed to simulate wireless medical device networks with, and without, QoS; and
5. Simulation of network performance with a tool such as CPNtool may be essential to predict and avoid life-threatening data delays.

Very little has been written about the need for QoS planning in healthcare applications. Failure to take network risks seriously is unnecessary and might even be taken as negligent behavior. The Institute of Medicine's 1998 report "To Err is Human" identifies tragic levels of medical errors in healthcare. Caution and careful, competent planning and design are needed to ensure that new wireless technologies do not introduce serious new risks in the healthcare environment.

One future direction for this work is to introduce more parameters and variables and create and analyze a hierarchical model using CPN.

REFERENCES

1. Briggs, B. (2004) Hospitals Bring Medical Devices into the I.T. Loop, Health Data Management, February.
2. Cohen, T. (2001) The convergence and integration of medical and information technology. Engineering in Physical Science and Medicine Annual Meeting. Fremantle, Australia.
3. Gehlot, V. (1988) Performance Specification and Livelock Detection/Correction of a Protocol Using Timed Petri Nets, Proc. International Conference on Communications, pages 1286-1290.
4. Gehlot, V. and Lee, I. (1988) Formal Specification and Analysis of DMI—An X.25 Based Protocol, Proc. IEEE INFOCOM'88, pages 641–650.
5. Gehlot V. and Sloane, E.B (2004) Applications of the Petri Net to Simulate and Validate Complex, Multi-Vendor, Heterogeneous Networks of Wireless Medical Device Systems, Proceedings of the Tenth Americas Conference on Information Systems, New York, NY, August 2004, pp. 4631-4636.
6. Health Consultants (2003) Patient-focused care boosts patient satisfaction while enhancing safety: wide variety of technology brings care directly to the patient, HealthCare Benchmarks and Quality Improvement, October.
7. Institute of Medicine. (1998) To err is human: building a safer health system Linda T. Kohn, Janet M. Corrigan, and Molla S. Donaldson, editors.
8. Jensen, K. (1997) Coloured Petri Nets. Basic Concepts, Analysis Methods and Practical Use. Volume 1, Basic Concepts. Monographs in Theoretical Computer Science, Springer-Verlag, 2nd corrected printing 1997.
9. Ratzer et al. (2003) CPN Tools for Editing, Simulating, and Analysing Coloured Petri Net, in Proceedings of Applications and Theory of Petri Nets 2003. Published as LNCS Volume 2679, Springer-Verlag, pp. 450 – 462.
10. Reisig, W. (1985) Petri Nets. Springer-Verlag, New York.
11. Sloane, E. B. (2001) The Rebirth of Clinical Engineering in the Information Age. Biomedical Instrumentation. March/April.
12. Sloane, E. B. (2002) eHealth: IT and the Reinvention of Healthcare. Association of Information Technology Professionals, Montgomery County, PA.
13. Wing, J.M., Woodcock, J., and Davies, J. (eds.) (1999) Formal Methods (FM'99). LNCS 1708 (Volume I) and 1709 (Volume II), Springer-Verlag.