Building Reliable MD PnP Systems

Mu Sun, Qixin Wang, and Lui Sha
Computer Science Department, University of Illinois at Urbana-Champaign
{musun, qwang4, lrs}@uiuc.edu

Abstract

We address two necessary issues needed for developing safe and reliable MD PnP systems: robust wireless networking and automated checking for component interoperability and reliability. First, the robustness of a wireless network can be improved by the use of DSSS-CDMA, which tradeoff throughput to achieve higher reliability and persistence of connections as needed. Second, automated checking of components interoperability and reliability can be partially addressed by the use of the *Integrated Framework for Assumptions and Dependencies* (IFAD) by specifying a machine-checkable encoding of component information, checking interface assumptions, and finding how various environmental changes can impact the system.

1. Introduction

There has long been the goal of improving the quality of life through machine assistance. In the *Operating Room* (OR) this is no exception. Today, there are many devices in the OR. Some perform safety critical tasks (e.g. devices that substitute vital organ functionalities), while others perform useful but non-critical operations (e.g. devices that archive data for later analysis). There is no doubt that interoperability amongst these devices is an important issue. However, due to the diverse number of manufacturers for different devices, having interoperability is challenging. Currently, medical personnel still need to manually manage the cooperation between these devices. Sometimes this leads to loss of human lives:

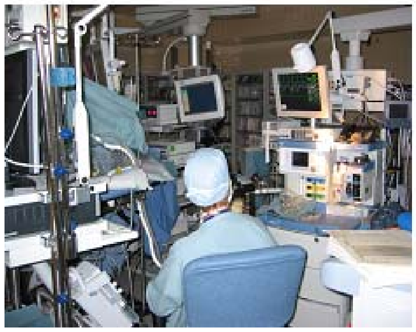
> *A 32-year-old woman had a laparoscopic cholecystectomy performed under general anesthesia. At the surgeon's request, a plane film x-ray was shot during a cholangiogram. The anesthesiologist stopped the ventilator for the film. The x-ray technician was unable to remove the film because of its position beneath the table. The anesthesiologist attempted to help her, but found it difficult because the gears on the table had jammed. Finally, the x-ray was removed, and the surgical procedure recommenced. At some point, the anesthesiologist glanced at the EKG and noticed severe bradycardia. He realized he had never restarted the ventilator. This patient ultimately expired.* [ASPF04]

This accident shows the need for a safe MD PnP environment. However, to provide the necessary degree of safety, a minimum set of features must be present:

1. A scripting language for the system configuration architecture is needed to standardize the description of component roles: the functional API, the data flow between them, and safety, security, and reliability requirements.
2. A verifiable safety-interlock protocol must be added to ensure the safe operation of a component despite other failures. In reference to the accident, a safety-interlock protocol should never allow the ventilator to be off for more that a specified number of seconds even if the network, x-ray machine, or the operator console fails.
3. An automated tool for assumption validation and dependency tracking is required to allow for machine checking of design issues such as incompatible interfaces and frail points in the architecture. For example, the tool shall ensure that the x-ray and ventilator communicate with the same protocol and shall evaluate how much of the ventilator functionality depends on other components.

4. A highly reliable wireless network to reduce the hazards of tripping over wires in an OR and to allow for easier PnP of devices.

Our research group has looked into all of these issues. In this paper, we discuss issues 3 and 4 in detail. As the first step, we emphasize the need for a reliable wireless network because before medical devices can cooperate seamlessly and reliably they must be connected together physically. Wires and cables can be used for these connections. However, this presents a hazardous work environment where medical personnel maneuvering around these cables and wires risk tripping over them (see **Figure 1** (a) for a wired OR). Healthcare professionals are explicitly requesting to "use wireless technologies to eliminate the 'malignant spaghetti' of cable clutter that interferes with patient care, creates hazards for the clinical staff and delays positioning and transport" [Goldman06] (see **Figure 1** (b) for a wireless OR). Nonetheless, one of the topics that must be addressed is how to make wireless connections reliable and robust enough for an OR environment so that wireless connections can replace most of wired connections. Fading due to obstructions, or accidental turning on of same *Radio Frequency* (RF) band wireless devices may interfere the wireless connections between medical devices. We solve with this problem by using *Direct Sequence Spread Spectrum Code Division Multiple Access* (DSSS-CDMA) technology which fully exploits the low-data-rate feature of the many OR communication routines. Specifically, for device control, vital sign monitoring and medical record retrieving, the typical data rates are below 10Kbps, usually around 1Kbps. This enables DSSS-CDMA wireless connections to achieve a 10~35dB gain on minimum jamming power over IEEE 802.11 schemes (the widely adopted wireless LAN standard). As an example, suppose the wireless communication uses the same RF band as an IEEE 802.11b router, the DSSS-CDMA wireless connection may sustain even while an IEEE 802.11b router is transmitting from a couple of meters away.


(a) A wired OR

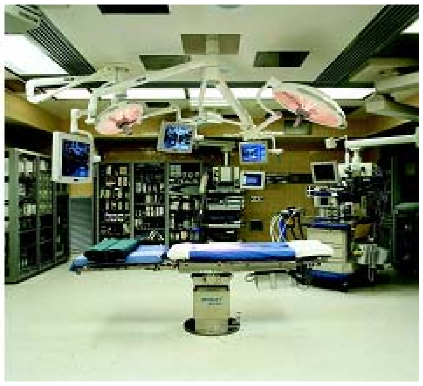

(b) A wireless OR

Figure 1 Wired and Wireless OR (pictures quoted from presentation by Dr. Julian Goldman at NSF High Confidence Medical Device Software and Systems workshop)

A different but related issue to connectivity is the problem of making sure devices behave correctly. Interoperating devices must share communication and computation resources. We must design the interoperable device architecture to prevent a safety critical device, for example the ventilator, from failing due to errors in a non-safety critical device, for example the x-ray machine. We must also give manufacturers a mechanism for synchronizing their interface designs with the devices created by many other manufacturers. Being able to have automatic verification of these architectures will definitely be necessary, as it is nearly impossible for humans to keep track of all the details of a large system of (potentially changing) devices. The first step to automation is to transfer the necessary component information buried in written documentation into a standardized machine encoding. The next step is to develop the software analyze certain aspects of the system. For example, the software checks whether the ventilator and x-ray uses the same communication protocol, and the software examines if a communication error between the ventilator and x-ray will cause the ventilator to fail. All of these issues are addressed by the *Integrated Framework for Assumptions and Dependencies* (IFAD) which is a merger of the *Assumptions Management Framework* (AMF) [Tirumala06] for checking interface assumptions between different devices and the *Dependency Management Framework* (DMF) [Ding06] for checking how errors and failures propagate among various devices.

The rest of this paper is organized as follows. Section 2 elaborates our proposal of a robust wireless LAN. Section 3 gives an example of a MD PnP system and elaborates on the functionality and usage of IFAD. Section 4 concludes with a summary of our work and our plans for the future.

2. Design of a Robust Wireless LAN

Although the demand to cut the wires and cables in the OR is strong, wide adoption of OR wireless network is hampered by the robustness concern. Unlike wireline, various adverse channel conditions, such as attenuation, multipath, and interference, may break wireless connections. A robust wireless network shall tolerate the attenuation and multipath fading caused by obstructions in the OR, and shall also tolerate common forms of *Radio Frequency* (RF) interference, such as inadvertently (or maliciously) turning on common RF devices such as cell phones, PDAs, wireless phones, microwave ovens, etc.

Main stream wireless LAN schemes, such as IEEE 802.11 (WiFi), deal with adverse channel conditions with backoff, i.e., defer wireless transmission till the channel condition recovers. This approach, however, fits poorly in the context of the OR, which mainly has to forward vital signs and device control traffics *continuously* in real-time. To find a more appropriate solution, we notice many of these traffics bare the feature of low data rate. This feature enables highly robust wireless connections if *Direct Sequence Spread Spectrum* (DSSS) modulation and *Code Division Multiple Access* (CDMA) medium access control schemes are used.

In DSSS-CDMA, every bit is multiplied (scrambled) with a *Pseudo Noise* (PN) sequence of g chips before sent out. At the receiver, if the same PN sequence with the same phase is multiplied (descrambled), the original bit recovers; otherwise, a Guassian noise is received[1]. The ratio between chip rate r_c and bit rate r_b is called *processing gain*, denoted as g (usually $g \in \mathbf{Z}^+$). Therefore every PN corresponds to a data channel. Several DSSS channels may coexist in parallel by using different PN sequences. Such medium access control scheme is called CDMA. DSSS-CDMA can be summarized by its Bit Error Rate (BER) upper-bound formula:

$$P_{BER} \le \exp\left(-\frac{gP_u}{\sum_{ch} P_{ch} + N + J} \right), \qquad \textbf{(1)}$$

where P_{BER} is the bit error rate; P_u and $\sum_{ch} P_{ch}$ denote the received power of the intended channel and all channels respectively, which are affected by nodes' maximum transmission power and wireless channel conditions (i.e. attenuation and multipath fading); N is the thermal noise power; and J represents the received power of all external interfering (a.k.a. *jamming*) noise. A wireless channel is more robust if it tolerates more jamming noise J.

[1] Interested readers can refer to [Wang05b] Appendix I for a more detailed tutorial on DSSS-CDMA.

We can regard $gP_u / (\sum P_{ch} + J)$ as the effective *Signal to Noise Ratio* (SNR). Inequality (1) shows the P_{BER} upper-bound decreases negative exponentially as the effective SNR increases. To maintain a wireless connection means to keep P_{BER} below a predefined threshold Θ_{BER}. Therefore, the higher the processing gain g, the more jamming noise J the intended wireless channel can tolerate (to keep P_{BER} below Θ_{BER}), which means the wireless channel is more robust. In other words, to maximize robustness means to maximize g. Since $g = r_c/r_b$, and usually chip rate r_c is fixed due to hardware and multipath limitations, maximizing g means minimizing the bit rate r_b. That is, by deploying slowest bit rate possible, we can maximize the robustness of wireless connections. Fortunately, the many vital sign and device control traffics in OR bare low data rate feature. By fully exploiting this feature, we can build significantly more robust wireless connections.

The above idea is elaborated and evaluated in our previous work [Wang05a], which carries out both specific demonstration and comprehensive Monte Carlo simulations to show the effectiveness of using DSSS to increase robustness.

In the demonstration, a central controller, two robots, and an RF inferring (jamming) node are placed at (0, 0), (-3, 2), (4, 0) and (7, 0) (meter) respectively. The controller must continuously communicate with the two robots over wireless to control them. Any interrupt of over 0.1sec causes the robot to malfunction. The jamming node transmits continuously for 10 seconds to interfere the wireless communication between the controller and the robots. All nodes transmit with the same power. The demonstration shows with DSSS-CDMA fully exploiting low data rate feature, the wireless links between the controller and robots survive the RF interference from the jamming node; while with IEEE 802.11, even at the most robust mode, the wireless connection breaks, causing robot malfunction.

The Monte Carlo simulation assumes a room of 20m × 20m, with a controller at the center of the room and n robots put across the room according to uniform random distribution. The controller controls each robot over a wireless connection. Each control packet is of 152 bits, and the minimum control rate is $f^{\min}$ Hz, i.e. one control packet must arrive at the robot every $1/f^{\min}$ second. Robustness is quantitatively measured by $J^{\min}$, the minimum RF interference power needed to breakdown at least one wireless connection.

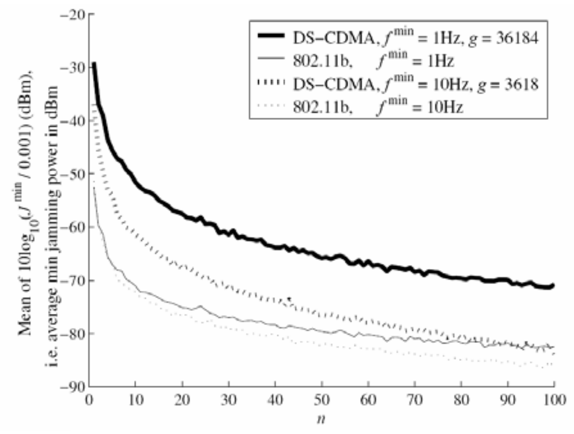
In each trial, we randomly generate a layout of the n robots and a wireless medium instance, and then compare robustness ($J^{\min}$) between the proposed DSSS-CDMA scheme and IEEE 802.11b/a, the most widely adopted wireless LAN schemes[2]. The wireless medium instances follow the model depicted in **Table 1**, number of connections (robots) n varies from 1 to 100, and minimum control rate $f^{\min}$ can be 1Hz or 10Hz. For each n and $f^{\min}$, 200 trials are simulated. **Figure 2** summarizes the simulation results.

Table 1 Wireless Medium Model

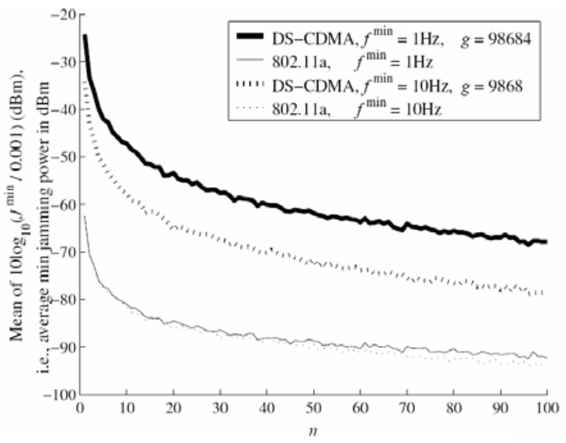
Large-scale path loss model	Log-normal shadowing model with β = 4~6, σ = 6.8dB *
Multipath fading model	Rayleigh
Multipath max excess day	90.909nsec
Additive white Gaussian noise **	Spectral density = -174dBm/Hz

* β is the path loss exponent, σ is the log-normal standard deviation.
** Typically refers to thermal noise.

[2] IEEE 802.11g is not explicitly considered since it is basically a union of IEEE 802.11b and IEEE 802.11a.



(a) Comparison with IEEE 802.11b



(b) Comparison with IEEE 802.11a

Figure 2 Robustness Comparison: robustness is measured by J^{min}(watt), the minimum jamming (i.e. external RF interference) power needed to break down at least one wireless connection. *n* is the number of wireless connections. Note the curves for the proposed DSSS-CDMA scheme are lower bounds for J^{min}, while the curves for IEEE 802.11b/a schemes are upper bounds.

According to **Figure 2**, our proposed DSSS-CDMA scheme obviously tolerates much more RF interference than corresponding IEEE 802.11 schemes. When f^{min} = 10Hz and 1Hz, our DSSS-CDMA scheme achieves approximately 10dB and 20dB improvement on robustness than IEEE 802.11b. Compared to IEEE 802.11a, the improvement on robustness is approximately 25dB and 35dB respectively[3]. These are significant results according to communication engineering standards. This is because of our drastic change of design philosophy, from aiming at high throughput to high robustness with low data rate.

3. Assumption Verification and Dependency Tracking

Over the years the number of components in a system has increased exponentially. Human checking of these systems has become nearly impossible, as almost no one can have a clear picture of the entire system in a large-scale project. This leads to an increased demand for automated tools to analyze these systems. Responding to this demand, we developed preliminary frameworks. The *Assumptions Management Framework* (AMF) [Ajay06] and *Dependency Management Framework* (DMF) [Ding06] developed by Ajay Tirumala and Hui Ding respectively (both alumni members of our research group). Here we describe an integrated framework of AMF and DMF, the *Integrated Framework for Assumptions and Dependencies* (IFAD), which allows for easier simultaneous access to both frameworks' functionalities and also automates interoperations between the two. In section 3.1, we start with a hypothetical MD PnP example. In section 3.2, we briefly cover the contributions of AMF and DMF. In section 3.3 we show how to encode different design aspects of the example system in a machine checkable format. In section 3.4, we show how to use IFAD and interpret the results.

[3] Note that for each setting, the tolerable jamming power decreases when the number of connections (*n*) increases. This complies with the Shannon bound of information theory [Cover91], which basically says when the application layer data throughput increases, and the signal power is given, the tolerable noise power decreases.

3.1 Example MD PnP Application

It seems fitting to show an example MD PnP system motivated by the real life situation in the introduction (see **Figure 3**).

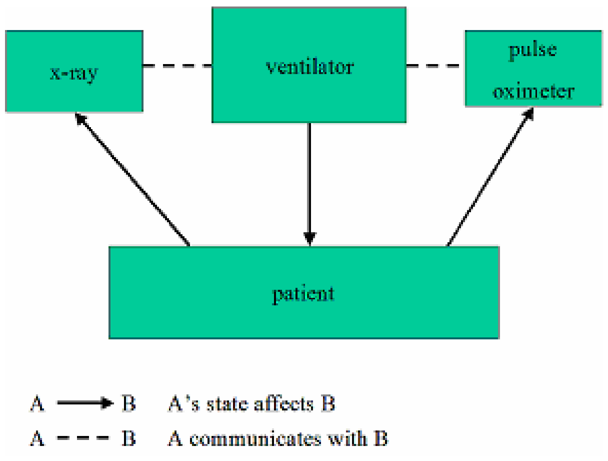


Figure 3 MD PnP Example

In this hypothetical example, the operator interface automates the job of medical personnel in terms of taking device readings and communicating between different devices. In addition to the MD PnP enabled ventilation machine and X-Ray machine, a pulse oximeter is added to the system to improve ventilation management. All these devices are connected by a reliable wireless network (with rare message drops). In order to guarantee a patient's safety in term of respiration, the ventilator is set to pause for at most *x* seconds in *y* minutes as specified by the experienced medical personnel. To improve ventilation support during the pause period, the oximeter periodically reads the patient's oxygen saturation level and sends it to the ventilator. If a patient's blood oxygen level drops below some threshold determined to have adverse health effects, the ventilation machine will resume before the expiration of *x*-seconds (this is used to optimize the patient's health management). Finally, upon receiving the command sent from the operator interface, the X-Ray machine will take the photo during the window of time in which the ventilator pauses.

3.2 Brief Overview of AMF and DMF

From a safety perspective, by ensuring that the MD PnP enabled ventilator will pause for no more than *x* seconds in *y* minutes as specified by medical personnel, the ventilator guarantees the safe respiration of the patient, even if the wireless network, the X-Ray machine, or the oximeter fails. From the perspective of dependency tracking, we say that X-Ray machine *depends* on the network and ventilation machine to take the photo. However, the ventilation machine's operation is independent of the operation of the X-Ray machine. Furthermore, the ventilation machine *uses* the network and the oximeter to optimize ventilation support, but does NOT depend on them for safe operation. The role of DMF is to encode the specified dependency requirements between components that are directly interacting and generate the system level dependency graph. It checks if dependency relations are well formed and analyzes potential paths of fault propagation.

In the discussion above, we have implicitly assumes that the MD PnP machines can exchange messages and work with each others successfully. However, this may or may not be the case when we assemble old and new devices from different manufacturers. A device can work correct within a particular configuration only if its assumptions regarding its external environment are satisfied. For example, European machines use metric while many US equipments use British units. If they are inadvertently mixed together in a configuration, such flaws should be automatically detected during configuration time. This is the task of Assumption management, and its functionality is available in AMF.

3.3 Encoding Assumptions and Dependencies

It should be noted that much of a device's assumption and dependency information is readily available in its written documentation. However, computers cannot readily process this information. One method is to have manufacturers or third party suppliers of these devices document the assumptions and dependencies in a machine-readable format.

Most constructs in IFAD are quite intuitive, but we will still present some formal definitions. The group of MD PnP devices forms a *system*. Each device in the system (x-ray, ventilator, pulse oximeter) is called a component. Each component has a *criticality level* associated with it describing the importance of that component. The ventilator has a high criticality level since the patient directly depends on it. The x-ray has a lower criticality level since the patient's life has no direct or immediate dependency on it. The pulse oximeter has a criticality level between the ventilator and x-ray since the patient's life does not directly depend on it, but it is used to help notify when a patient's health is adversely impacted. A component can make *assumptions* on other components. The ventilator makes the assumption that the pulse oximeter provides its readings in certain units (a percent between 0 and 100 or a fraction between 0 and 1). On the other end, a component can provide *guarantees* to other components. The pulse oximeter provides a guarantee to the ventilator that its readings are given as a percentage. Each component also has a set of potential *faults* or errors. The pulse oximeter can have errors such as incorrect reading, communication error, out of power, etc. An error in one component can *cause* errors in other components. For example if the ventilator fails to pause then the x-ray would have blurry image. We now take a look at how the definition for the ventilator component looks like[4]:

```
Component ventilator
   criticality: 10
   faults: noTurnOff, noTurnOn, xRayCommError, pulseOximeterCommError
   assumes pulseOximeter
      readingRange: [0,1]
         impact: xRayCommError, pulseOximeterCommError
   assumes enironment
      interferenceNoise: < -60dBm
         impact commError
   guarantees xRay
      turnOffTime: 10 seconds
```

This defines the ventilator component. First, it is assigned a criticality of 10. Criticality can be any arbitrary positive integer with more critical components having larger integer values, so it is a relative measure determined by the system designer. The next line lists the possible faults for the ventilator: it might fail to turn off, fail turn on, or have communication problems with the x-ray or pulse oximeter. The next block describes the assumptions that the ventilator makes on the pulse oximeter. The ventilator assumes that the pulse oximeter provides readings as a fraction or the range of values is between 0 and 1. If this assumption does not hold (or is *violated*) then the impact on the ventilator will be communication

[4] To prevent syntactic distractions, the code in this section will be presented as intuitive pseudo code. The detailed grammar specifications of both AMF [Tirumala06] and DMF [Ding06] can be found on their corresponding papers.

problem with the pulse oximeter. Next, the ventilator assumes that the noise in the environment is below a certain threshold. Otherwise there will be a communication problem between all the components. Notice that environment is treated as a component in the system. Any logical unit in a system can be a component as decided by the system designer (even the patient can be thought of component). In the last statement block the ventilator provides a guarantee to the x-ray that it will turn off for 10 seconds (when the x-ray needs to take a picture). Now that the basic syntax has been described through example here are some definitions for the other components:

```
Component xRay
   criticality: 1
   faults: jammed, blurryImage, commError
   assumes ventilator
      turnOffTime: >= 10 seconds
         impact: blurryImage
Component pulseOximeter
   criticality: 5
   faults: wrongReading, commError
   guarantees *
      readingRange: [0,100]
Component environment
   guarantees *
      interferenceNoise: -100dBm
```

The only new construct here is the `guarantees` clause in the pulse oximeter. The manufacturers of the pulse oximeter design the oxygen reading to be used by many components other than the ventilator. All of them will need to know what units or range the reading is providing. The asterix (`*`) means that a guarantee is provided to all components that use its output readings (including the ventilator). The environment also uses this construct since all components may be affected by interference noise. The criticality is omitted from the environment (an arbitrary high value may be assigned since everything depends on it). As the example shows, all assumptions are matched by a guarantee, if this is not the case, an *Assumption Guarantee Matcher* will give a notification of unmatched assumptions and guarantees. Many details are left out to avoid being distracting from the important points. For example the x-ray and pulse oximeter are also affected by the `interferenceNoise`. Also, if power is an issue, components such as power generator and back up generator may also be added.

Besides component definitions we also need to specify fault propagation rules. The notation `ventilator.noTurnOff` means the fault `noTurnOff` of component `ventilator` (i.e when the ventilator is unable to turn off). Also, `ventilator.noTurnOff -> xRay.blurryImage` means that the fault `noTurnOff` in `ventilator` will cause the fault `blurryImage` in `xRay`. Here are some fault propagation rules for the components:

```
ventilator.noTurnOff -> xRay.blurryImage
pulseOximeter.wrongReading -> pulseOximeter.commError
pulseOximeter.commError -> ventilator.pulseOximeterCommError
xRay.jammed -> xRay.commError
xRay.commError -> ventilator.xRayCommError
```

Notice that Boolean statements can also be used to specify how a combination of faults can cause a certain fault to occur. Now that these assumptions and dependencies are encoded, we show how automated checking of this data can help evaluate the robustness of a system.

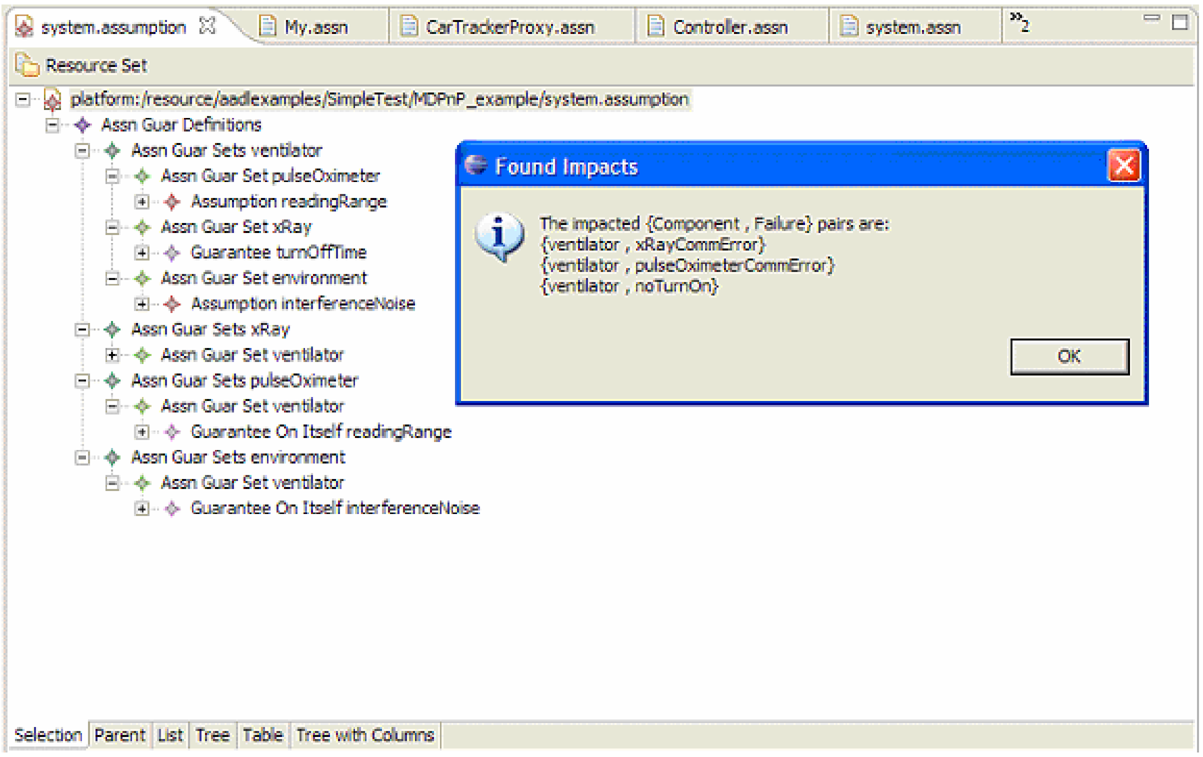
3.3 Using IFAD

IFAD is developed as a plug-in to Eclipse (http://www.eclipse.org). IFAD takes many of AMF's user input functionalities such as describing components using a graphical XML editor. If we input the above component and dependency descriptions, normal AMF functionalities can be applied to it. Checking for violated assumptions will report that the assumption `readingRange` made by `ventilator` on `pulseOximeter` was violated. By reporting these assumption violations, accidents due to interface incompatibilities can be prevented (if the ventilator did not interpret the pulse oximeter readings correctly, it may not realize when the patient is in a critical state). Normal DMF's functionalities can also be used. A fault propagation check for `wrongReading` in the `pulseOximeter` will result in the `ventilator` having `pulseOximeterCommError`. This can be deduced by looking at the dependency encoding. However, in a typical system of tens or hundreds of components, machine checking is the only way to find all propagation paths of a fault.

Besides providing the base functionality IFAD allows the user to find how an assumption violation impacts the entire system. If the ventilator pause time was set incorrectly, the ventilator will now have a communication dependency:

```
ventilator.xRayCommError AND ventilator.pulseOximeterCommError ->
ventilator.noTurnOn
```

Assuming no other assumption violations, if we simulate intentional malicious interference to the system by setting the interference noise to be -40dB, running *find impact* on this example we obtain the results in a message popup (**Figure 4**). IFAD shows the ventilator experiences the noTunOn fault. The design of the system is shown to be unsafe as the ventilator no longer turns on when strong interference is present. In this way, IFAD can measure the robustness of the system under to a certain environmental changes. After encoding the behavior of all the devices in an OR, one can test how various environmental factors (temperature, EM interference, humidity, etc.) affect the system.



The impacted {Component , Failure} pairs are:
{ventilator , xRayCommError}
{ventilator , pulseOximeterCommError}
{ventilator , noTurnOn}

Figure 4 Screen Shot

There are numerous other possible functionalities in IFAD (some of which are still being developed). For example, one can query to see which assumptions are critical to the prevention of certain faults or find the criticality of an assumption by finding the most critical component affected.

4. Conclusion and Future Work

In this paper we have addressed two necessary issues needed for developing safe and reliable MD PnP systems: robust wireless networking and automated checking for component interoperability and reliability. The robustness of a wireless network can be improved by the use of DSSS-CDMA, which exploits slow data rates to achieve higher reliability and persistence of connections. Automated checking of components interoperability and reliability can be partially addressed by the use of IFAD by specifying a machine-checkable encoding of component information, checking interface assumptions, and find how various environmental changes can impact the system. However, it should be noted that IFAD cannot guarantee the safety and interoperability of a system, because the information input by users regarding assumption and dependency may contain errors.

Nevertheless, IFAD is a useful tool to help the system architects to evaluate and improve the design. After full development, we plan to annex IFAD into the *The Architecture Analysis and Design Language* (AADL) (http://www.aadl.info/), a standard already used widely for encoding system component information. This will allow for easier communication between device manufacturers and allow for the design of safer systems overall. We are also working on a combined WAN and wireless LAN solution for medical MD PnP to extend future interoperability to wide area network and even Internet.

References

[ASPF04] ASPF Newsletter. Volume 19, No. 4, 41-60. Winter 2004-2005.

[Cover91] T. M. Cover and J. A. Thomas, Elements of Information Theory. Wiley-Interscience, 1991.

[Ding06] Hui Ding, Leon Arber, Lui Sha, and Marco Caccamo, “The Dependency Management Framework: A Case Study of the ION CubeSat”, in Proc. of 18th IEEE Euromicro Conference on Real-Time Systems (ECRTS’06), 2006.

[Goldman06] Julian M. Goldman, "Medical Device Connectivit for Improving Safety and Efficiency", in American Society of Anesthesiologists Newsletter, vol 70, number 5, May, 2006.

[Tirumala06] Ajay Tirumala, “An Assumptions Management Framework For Systems Software”, PhD Thesis for UIUC CS department, September 2006.

[Wang05a] Qixin Wang, Xue Liu, Weiqun Chen, Wenbo He, and Marco Caccamo, "Building Robust Wireless LAN for Industrial Control with DSSS-CDMA Cellphone Network Paradigm", in Proc. of 26th IEEE International Real-Time Systems Symposium (RTSS 2005), Dec., 2005. pp 3-14.

[Wang05b] Q. Wang, X. Liu, W. Chen, W. He, and M. Caccamo, Technical report on building robust wireless lan for industrial control with dsss-cdma cellphone network paradigm. *http://www-rtsl.cs.uiuc.edu/papers/dsss cdma tr.pdf*, 2005.