HUMAN AND WORKFLOW ISSUES WITH SMART SENSOR NETWORKS

Mac Reiter
Nomadics, Inc.
Stillwater, Oklahoma
reiter@nomadics.com

Charles H. Jones, PhD
Edwards Air Force Base, California
charles.jones@edwards.af.mil

ABSTRACT

This paper presents methods of using standard PDAs, smart sensors, XML-based network and user interface descriptions, and graphical sensor network management to provide an installation-to-use workflow. Techniques discussed will include: methods of providing additional capabilities to PDAs, methods of automatically constructing user interfaces based on constraints and requirements from both the sensor descriptions and the PDA capabilities description, and methods of providing more natural selection of sensors for test setup.

INTRODUCTION

With the implementation of smart sensors (e.g., IEEE 1451 compliant sensors) there will be a shift in the way humans interact with sensors. Hopefully, this shift will lead to greater efficiencies in workflow. However, to fully realize the benefits, it will be necessary to implement new tools that take advantage of the intelligence being placed with the sensors. There are fundamentally two aspects of sensor interaction. The first is when installing or maintaining the transducer, in which case there is fairly direct access to a single sensor. The second is when dealing with the test suite of sensors as a whole. With the use of smart sensors, the second aspect includes the issue of how to manage data obtained from self-describing networks.

The use of Personal Digital Assistants (PDAs) seems an obvious choice for directly interacting with sensors. However, there are both benefits and limitations of using PDAs. One particular issue is how to get information between the PDA and more powerful workstations. The use of graphical user interfaces (GUIs) for managing data regarding the entire suite of sensors also seems an obvious choice. However, the background processing to provide these GUIs can be extensive. This requires internal data storage mechanisms as well as automated GUI generation. All of these issues are discussed in this paper.

USE OF PERSONAL DIGITAL ASSISTANTS (PDA)

If a user is installing or maintaining a sensor on a test vehicle or factory floor, the use of a handheld device seems appropriate. The following reasons make the use of a PDA a logical choice for such a handheld device.

1. Portability
2. Graphical Quality
3. Ease of user interaction
4. Networking abilities
5. Availability
6. Economics

PDAs are designed, first and foremost, to be easy to carry and use as the owner goes about their daily tasks. PDAs can easily be stored in pockets or on belt clips. Interaction is performed through touch screens, using a stylus that is stored inside the PDA itself. PDAs provide daylight-readable, high-contrast, full color screens with reasonable resolution. Today, 240 by 320 pixels is common, with 480 by 640 beginning to appear. PDAs power up to a usable state in a second or less, allowing the user to carry the PDA powered off most of the time, which extends battery life tremendously. While not ruggedized, modern PDAs are constructed to withstand everyday bumps and bangs without developing problems. Wireless connectivity is becoming widespread in modern PDAs. This gives the PDA access to huge amounts of information and support services through network connections to other machines. Finally, PDAs are ubiquitous and relatively inexpensive. Local retailers and online stores sell hundreds of different models typically for less than $500.

On the other hand, the small size of PDAs also forces limitations. Some concerns include:

1. Hardware interconnection
2. Processing power
3. Memory
4. Screen size

One of the biggest problems with using PDAs as a portable device for communicating with sensors is the variability of hardware interconnection options. Each new model generally changes its system interconnect. Even the standard connection ports change from year to year. Supported expansion ports include PCMCIA, CompactFlash, Secure Digital (SD), and MultiMedia Card (MMC). Unfortunately, each PDA typically only supports one of these, and the support may disappear in the following model.

PDAs have physical limitations to the amount of processing power that they can provide. Modern PDAs run at 400MHz or faster, but this is still an order of magnitude behind desktop personal computers (PCs). A battery-powered, handheld device simply cannot use the amount of electrical power necessary to compete with desktop PCs. The battery would be drained almost immediately. Even if some other power method was available, a handheld device cannot dissipate as much waste heat as a desktop PC that uses large heat sinks and high-volume fans.

Similarly, the size of a PDA limits the amount of memory and the size of the screen. Although some PDAs now have 64 MB of memory with expansion slots for memory sticks, they simply will never keep up with the storage available on a standard PC. Likewise, the roughly 3”x3” displays will never compete with the real estate on a standard monitor.

WIRELESS NETWORKING CAN MITIGATE PDA LIMITATIONS

The first limitation that needs to be addressed is the hardware interconnection. If the PDA cannot communicate with the sensor or sensor network, issues of storage space and processing power are immaterial. Fortunately, all PDAs created in the last several years, and all PDAs for the foreseeable future, support InfraRed Data Association (IrDA). While nothing is guaranteed in commercial-off-the-shelf (COTS) equipment, IrDA support is so prevalent, so easy to include, and so useful to consumers that it should remain available for years to come.

To take advantage of this connection, the interface device must become a standalone unit. Rather than plugging into the PDA and drawing power from its bus, it generally will need to supply its own power. An interface device for communicating with a sensor network may be able to draw power from the sensor network itself, but a device that communicates with a single transducer will need to provide power itself. This is the only significant change to the design model of the interface converter. It will still be responsible for converting commands from one bus – IrDA – to another bus – IEEE 1451.3, for example.

Remote Data Access

The next solution addresses the storage limitations of the PDA. Wireless access to a remote database allows the PDA to provide access to essentially unlimited data storage. The user application is still developed for the PDA, deployed on the PDA, and runs on the PDA, with only the data storage moved to the server side.

This solution provides a straightforward development effort, with a standard application that simply retrieves data from a remote server rather than a local database. It remains limited in terms of processing power, which may be a problem in some situations.

Wireless support can be used to take advantage of the current state of Internet and World Wide Web (or just Web) technology. Web browsers have always provided document retrieval and viewing, but modern browsers can provide full user interfaces through dynamic Web pages. Entire database management systems can be implemented entirely through logic held on a server and supplied to generic browser applications on arbitrary devices.

This solution provides significant capability to the PDA user in the field. Applications and data never need to be synchronized or updated, because they reside on a server. The only necessary application on the PDA is standard Web browser, which is supplied as part of the built-in utilities. Modern Web developers are fluent in this style of development, providing new application support as needed.

This solution is limited to interaction methods for which the Web was designed. The interaction needs to be primarily based on forms and data entry, with submissions, queries, and reports. Some deviation from this model can be tolerated, but a wholesale departure from this model requires a different solution. One example of such a requirement would be interactively viewing a three-dimensional model. Some systems exist to supply this capability, but they all require the client device – the PDA in this case – to perform the necessary processing for manipulating the model. The PDA typically does not have enough processing power for this task.

Remote Application Control

With the increased availability of high-speed wireless networks, it is now reasonable to provide remote control of applications running on desktop computers in other locations. This approach provides tremendous capability to the user in the field. As long as the PDA is capable of providing reasonable screen updates from the networked connection, it is powerful enough to do anything that can be done on a regular PC. This approach has been used for various projects, and it has been found that modern PDAs have more than sufficient capability to provide full-screen remote control of fully interactive multimedia applications over standard 802.11b 11Mbit/sec links. The actual transmission rate was limited by the PDAs processor, not the network link, and also provides very usable performance at 2Mbit/sec. Streaming video and interactive three-dimensional graphics can all be controlled remotely with this system. Experiments used the freely available, open source Virtual Network Computer (VNC) system. Developed PDA client software improved efficiency while leaving the PC server software unchanged.

The remote control approach can also be used piecemeal. Rather than providing remote control of an entire application, the application can be split between the PDA and the PC. Simple, conventional graphical user interface (GUI) components – buttons, text entry fields, list boxes, and similar controls – can be part of the application running directly on the PDA. When a portion of the GUI requires additional processing power, that portion can be generated by a remote PC, and displayed in the appropriate portion of the screen on the PDA. By splitting the work this way, the latency and responsiveness of the application are improved, while the network bandwidth required is decreased.

Security

The introduction of wireless networking creates security concerns in some environments. One solution is to use the open source implementation of Secure SHell (http://www.openssh.com/) on the PC server and the PocketPuTTY (http://pocketputty.duxy.net/) SSH (Secure SHell) client for the PDA. These tools are regularly reviewed by security experts, updated by their creators, and generally provide excellent data-in-flight security.

SSH provides secure communication tunneling. After establishing a secure connection to a PC, the PDA can create separate connection streams within that secure connection. It can then communicate with databases, Web servers, or remote control servers with high quality encryption. The tunneling is transparent to the communication protocols, allowing existing applications and software libraries to be used without modification.

AUTOMATIC GRAPHICAL USER INTERFACE (GUI) CREATION

One of the largest challenges with self-describing systems is the need for flexibility in user interaction. As the devices that users interact with gain the ability to describe themselves, applications must gain the ability to respond to those descriptions rather than simply using rigid interactions defined and implemented by the system designer.

A system needs to be created for going directly from a description of the data that is needed and available to a GUI that displays and interacts with that data. This system will initially be targeted at field users who are installing and configuring transducers. This system should provide access to the necessary fields for configuring the transducer, without cluttering the interface with unnecessary data. But the system also needs to gather all of the data in the transducer's Transducer Electronic Data Sheet (TEDS), so that the system can report this data to an Instrumentation Support System (ISS) after installation is finished. The automatic GUI creation will also be useful for desktop PC users who need to access configuration information about the transducers in their sensor network.

In the general case, the semantics of user interaction are very hard to represent. At a very basic level, will a particular data field be best represented by:

- Radio buttons : a very small number of mutually exclusive choices
- Check boxes : a very small number of simultaneous choices
- List boxes : a larger number of either mutually exclusive or simultaneous choices
- Spinner : numeric entry with a known step size
- Text : numeric entry of arbitrary step size

At the next level of complexity, issues of data field interdependence are encountered. A field specifying the number of channels in the transducer will affect the options presented later in a field for specifying which channels are in a channel group. As another example, the enumeration for a channel's data model – signed or unsigned, integer or float, and bit length – could affect legal values in fields that represent minimum or maximum values.

At the highest level of complexity, entries in one field can completely change other portions of the interface. The number of channels will modify the number of data entry panels needed. A field that contains a type code for a channel may change which fields are present in that channel's data entry panel completely. Actuators and sensors have different settings. Similarly, single event and sequence transducers require different types of settings.

A full TEDS contains quite a bit of information. Some of this information is of considerable importance to all users. Most of the information, however, is only relevant to particular users. Units, ranges, and locations are important to test engineers. Timing information is primarily useful to signal conditioners and analog-to-digital Converters (ADCs). Sampling rates cross the boundary, as they are primarily of interest to signal conditioners and ADCs, but are sometimes useful to test engineers when choosing between sensors. This leads to a concept of a "User Perspective," which allows grouping of data and controls based on which items are most useful to that kind of user.

An automatically created GUI needs some way of grouping this data so that the groupings are relevant and coherent. Related fields need to be together, and unrelated fields need to be clearly separated. On top of this requirement, the GUI should cleanly support the different user perspectives described previously. This may require some duplication of data fields and controls, because individual fields may be useful to various users. Sampling rate was an example of the kinds of data fields that can cross user perspectives.

DATA REPRESENTATION

The intention is to create a hierarchical set of eXtensible Markup Language (XML) schemas that will provide a way of encoding a TEDS layout. These XML Schemas can then be used to create one XML Schema for each IEEE 1451 point standard. The automatic GUI generator will then create a GUI for manipulating data sets that are constrained by a specific IEEE 1451 TEDS XML Schema.

Because XML Schemas are used at several levels, a more structured breakdown may help understand the intended hierarchical structure:

1. Data Types Schema(s) – XML Schema(s) that describe how a particular data type can be encoded. This level would handle issues of enumerations, integers versus floats, signedness, and bit length. Data Type Schemas provide information that is needed for the lowest level decisions involving graphical controls and data storage.
2. Table Schemas – XML Schemas that use the Data Type Schemas to define how a single TEDS table can be encoded. For example, the IEEE 1451.2 Channel Identification TEDS would have a Table Schema that defined its data fields in terms of Data Types Schemas. Table Schemas provide information that is useful for grouping related data fields, as well as supporting user perspectives. Each field will need to list the user perspectives for which it is useful, along with a ranking of its importance to that user perspective.
3. TEDS Schemas – XML Schemas that collect Table Schemas and encapsulate knowledge about how information in one table may affect the presence or number of other tables. For example, the entire IEEE 1451.2 TEDS collection would be a single TEDS Schema. TEDS Schemas encode the highest level semantics about inter-table relationships. This is the input to the automatic GUI creator.
4. TEDS File – An XML document, using the appropriate TEDS Schema, which describes a single transducer. For example, an IEEE 1451.2 temperature sensor would have a single TEDS File, which must be valid under the IEEE 1451.2 TEDS Schema. The resulting GUI is responsible for manipulating these files.

The automatic GUI creator will use the information in the TEDS Schema to construct a GUI that is capable of gathering and representing the information needed for a TEDS File.

SENSOR SET MANAGEMENT

Once a set of transducers has been configured and installed into a sensor network, test engineers need to be able to manage all of the information related to that network. IEEE 1451 standards help gather all of the necessary information, but some tools are still needed to manage information overload. The test engineer needs to be able to quickly and easily locate the sensors needed for a test, without referring to obscure sensor ID numbers and installation logs.

Intuitive Queries

Treating all of the information about a sensor network as a database is fairly common. The ability to issue queries against that database is then taken for granted. Unfortunately, the queries are typically expressed in something like the Structured Query Language (SQL), which is not the most natural expression of intent for a test engineer. The largest problem with such queries is their all-at-once nature. To retrieve an exact set of information, a query must be constructed such that it retrieves all of the information desired, and none of the information that is not desired. Constructing such a query can be difficult, and sometimes impossible.

In-house usage experiments on other projects has suggested that iterative refinement is much easier to use for constructing sets of information. Many database tools do provide query refinement tools, which basically allow the user to add AND or OR clauses to the query. The intent is to experiment both with other methods of refinement and with other methods of reporting. The hope is that this combination will provide a faster and more intuitive interface for the test engineer.

Rather than directly adding and removing AND or OR clauses from a query, a multi-stage selection may simplify the process. One region represents the entire selected set. Another section of the GUI allows the user to specify conditions that should be added to the set. Matching sensors are placed in a holding pen. Yet another section of the GUI allows the user to specify conditions that should not be included in the set. Matching sensors are removed from any set that is in the holding pen. Note that neither addition nor subtraction occurs directly to the full selection set. The user can also simply drag individual sensors into or out of the holding pen, if they are fairly close. When everything in the holding pen belongs in the main set, they can add the holding pen contents to the main set and start a new holding pen. Technically, this is identical to creating AND and OR clauses, where addition is an OR clause and subtraction is an AND NOT clause. The staged presentation serves to simplify and visualize the nesting of logic clauses.

During early stages of refinement, the response set may be so large that it overwhelms the user. Some method of providing higher-level information about the nature of the currently selected set of sensors may help the user more rapidly tune the set of selected sensors. Thoughts at this time include clustering diagrams and some statistical feedback. The higher-level feedback should be graphical when possible, to enable easier and faster understanding of the system.

Graphical Selection

An important aspect of test setup involves the location of the sensors. Textual descriptions of locations tend to have discretization issues. If the locations are too broad – “Left Wing” -- an engineer that needs to study behavior around a particular portion of the wing would have to use other methods to remove all of the sensors that were on inappropriate portions of the wing. At the other extreme, if the locations are too narrow – “Panel 38” -- an engineer that needs to study behavior over an entire section of the craft must include several locations, which is an error prone task.

By storing location information in the user application during installation, the system will allow the test engineer to select sensors directly by location, with whatever degree of precision needed. The system can provide controls for specifying location either through three-dimensional models or through a simpler pair of images. The pair of images would provide one top-down view and one sideways view. Between the two images, locations and volumes in three-dimensional space can be specified. These techniques are used in computer-aided drafting (CAD) and computer-based modeling, and are quite effective. Using the methods described previously, these controls can be presented on the PDA during installation so that the user can specify where the sensor has been installed. The test engineer can then select sensors by simply surrounding the volume of interest.

SUMMARY

In order to fully realize the benefits of emerging smart sensor technology, it is necessary to develop new tools to interact with the sensors. The use of PDAs as a handheld tool for direct interaction with individual sensors along with GUIs for management of sensor suites are obviously choices. The interaction between the PDA and PC needs some careful design but the use of XML for data exchange will help considerably. The most difficult part of developing these tools will be automating the GUI creation so that each suite of sensors is displayed and managed in its own unique fashion.

ACKNOWLEDGEMENTS

This paper is related to work being supported by Air Force Flight Test Center under Small Business Innovation Research (SBIR) contract number FA9300-04-M-1012.