DUST NETWORKS

Sukun Kim
Rufus McLain

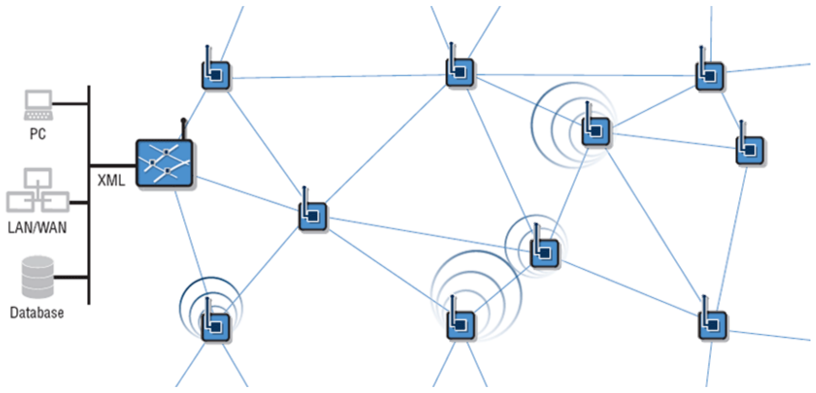
MBA 290/ENG 298A – Introduction to the MOT

Wireless Sensor Networks – An Introduction

Advances in semiconductor technology following Moor's law have enabled enormous growth in information technology: PC, Internet, wireless phone, etc. Technology is now passing a point where a battery-powered computer is powerful enough to operate wireless communication devices for an extended period of time. As a result, wireless sensor networks (WSN) have been developed in universities and research labs, paving the way for companies like Dust Networks. The following paper looks at the market for WSN and analyzes ways in which Dust Networks will be able to thrive in this new and growing environment.

Dust Networks & Wireless Sensor Networks

Founded in 2002 by Kris Pister and Rob Conant, Dust Networks provides low-powered wireless sensor networks to solution providers,



integrators, and OEMs for monitoring and controlling applications. (See Exhibit 1 for a history of the Dust Networks) A wireless sensor network is a wireless network of small computers with sensors and actuators. These small computers are called motes and they are composed of three principle parts: a senor or actuator, a microprocessor, and a radio chip. The sensor reads physical data of the real world, the actuator can control basic functions like turning off a light or changing a room temperature, the microcontroller processes data from the sensor and makes decisions, and the radio chip communicates data wirelessly. The next element of the network is the SmartManager that collects all the information from the motes and acts as a gateway between them and the server. Throughout the network and on the server there is software that executes routing, timing, network optimization and management functions to ensure that connectivity and long-term reliability. (See Exhibit 2 for a screen shot of the software)

History of WSN Technology

Wireless Sensor Networks started as a DARPA (The Defense Advanced Research Projects Agency) sponsored project named NEST at UC Berkeley in 2000. NEST (Network Embedded Systems Technology) is used for unmanned monitoring of enemy battlefields. Unmanned air vehicles would drop motes that would use magnetometers and sounder to detect enemy movement. These networks would be able to determine the type, speed, and direction of ground vehicles. Soon after the success of this project, Intel saw the potential of this technology and hired, then DARPA manager, David Tennenhouse as a research manager. Intel then funded a company called Crossbow to work with UC Berkeley to further develop WSN technology. In 2002, Kris took leave from the research group at UC Berkeley to start Dust Networks. Since this time there have been many successful implementations of Wireless Sensor Networks.

Practical Applications of the Technology

<u>Machine Monitoring</u>

Imagine a motorcycle company and that uses manufacturing plant with a conveyor system and machines with bearings which wear out over time. When the bearings completely wear out or fail, the machine stops and the entire conveyor stops as well. It takes between two and three hours to repair the system and the entire downtime costs hundreds of thousands of dollars. However, by monitoring the vibrations of the machines, the company could determine whether a bearing is almost worn out and schedule the bearing replacement in advance during a routine downtime. One possible way to monitor vibration is by attaching accelerometers to machines and then wiring them to a central control room. However, wiring one accelerometer costs around one thousand dollars and if hundreds of machines need to be monitored, this cost becomes very high. A wireless sensor network could be used to solve this problem at a fraction of the cost.

<u>Seismic Testing</u> ($200 vs. $5,000 per node)

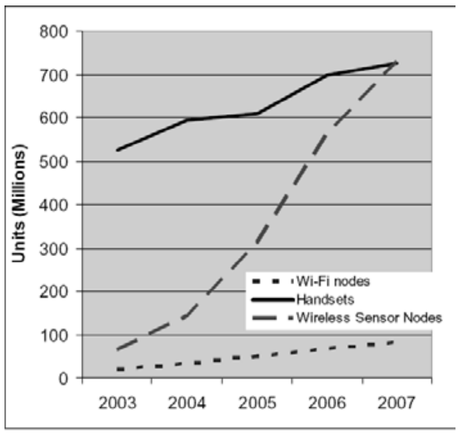
In the department of civil and environmental engineering at UC Berkeley, Professor Steve Glaser performed an experiment to look at the effect of an earthquake on a structure. A three-story model of building was excited on a shaking table to simulate an earthquake. Vibrations measured before, during, and after an earthquake could be used to determine the structural integrity of the building and pinpoint where the building was damaged. Before using wireless networks, the experiment to much longer because the time needed for wiring and the cost per node was $5,000, compared with $200 for wireless nodes.

<u>HVAC Energy Testing Nodes</u> ($100 vs. $500 per node)

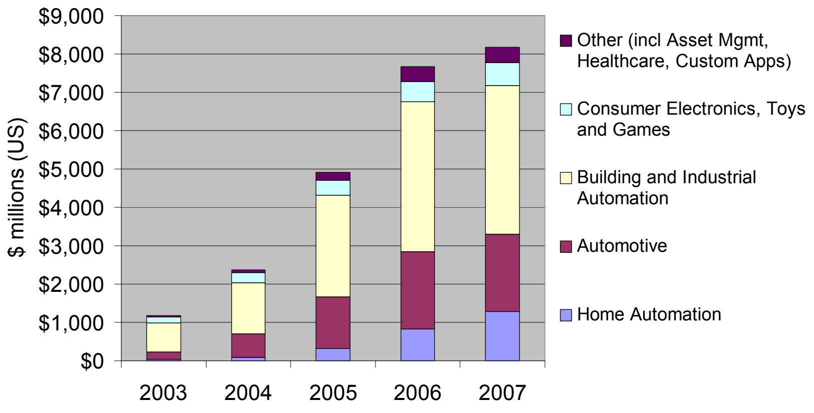
Dust Networks partnered with Honeywell on a project to install wireless sensor networks in a chain of grocery stores. These networks monitored electricity usage from lighting and temperature controls in an effort to analyze ways to save money. The wireless motes cost $100 verses the $500 per sensing point with a conventional system. The installation was also done in only three hours, versus three to four days.

Market Size

According to the Wireless Research Group, the market size for wireless sensor networks will be approximately $5billion this year and over $8billion in 2007. The number of units is expected to grow to just over 700 million by 2007, catching up with the number of wireless handsets. Building and industrial automation, two of Dust Networks' primary application foci, will constitute more than have of the total market.



Source: InStat/MDR 11/2003 (Wireless); Wireless Data Research Group 2003; InStat/MDR 7/2004 (Handsets)



Value in System Integration and Support Services (Module 4)

There are three layers in the value chain for the wireless sensor network market; component manufacturing, network manufacturing, and system integration. There are established players in each of these levels and different potentials for current and future economic value. In order for Dust Networks to capture the largest amount of this value as possible, they must correctly position themselves within this value chain. (See Exhibit 3 for a diagram of the value chain and the major players)

<u>Component Manufacturing</u>

The first layer of the value chain is comprised of companies that build the components that make up the motes in a WSN. The two major parts that these companies manufacture are the microcontrollers and the radio chip. The microcontroller is basically a one-chip computer made-up from a CPU, memory, and flash memory. The radio chip allows wireless communication between the different parts of the WSN. The three primary companies building microcontrollers are; Texas Instruments, Atmel, and Motorola. The two largest radio chip manufacturers are Chipcon and Freescale. The current trend within this market is to put both the microcontroller and the radio onto a single chip which has lower power consumption and lower cost. This trend blurs distinction between the two different types of manufacturers, making this market a free-for-all.

Network Manufacturing

Within the next level of the value chain are companies that take the individual components and produce motes, managers, and software for WSN. Along with Dust Networks, the other major players within this segment are Crossbow and Ember.

Crossbow (www.xbow.com) was founded in 1995 as a sensor manufacturing company (MEMS accelerometers, Gyros, etc.). In 2003 the revenue for Crossbow was $16M, double that of 2002. The software they provide for their WSNs is a stabilized version of TinyOS, similar to RedHat Linux. TinyOS are made up of many components and are constantly evolving. Crossbow bundles these TinyOS with hardware to create there WSNs. In addition, Crossbow holds both introductory seminars to attract new customers and advanced tutorial seminars to educate existing customers on new features and functionality. These seminars increase their customer base and bring in a substantial amount of revenue into the company.

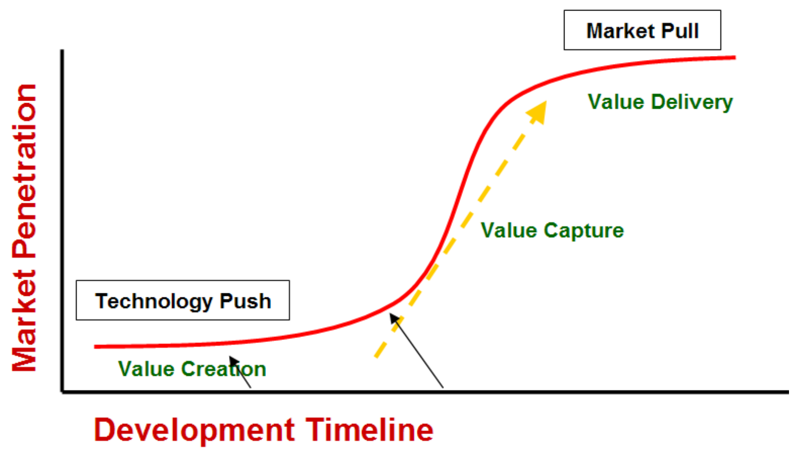
Ember (www.ember.com) is a Boston based company founded in 2001. Like Dust Networks, they are dedicated to the WSN market and provide proprietary software. They are also quick to adopt hardware and protocol standards and have entered a partnership with Atmel for the Zigbee networking and microcontroller platform. (the new standard for personal area network on top of IEEE 802.15.4)

System Integration and Support

At the top of the value chain are the conventional system integrators. These companies work directly with the end-users of the WSNs and handle everything from installation to system integration to support of the WSN. As a result, they produce the majority of the value and profit within the value chain. The major system integrators are large companies like Honeywell, Carrier, and General Electric.

Dust Network's Positioning within the Value Chain

Currently the market for WSN is in the early stage of the development timeline. The market penetration of this technology is relatively small and a good percentage of the total market value is realized in the creation of these networks. As they continue to push the technology forward they will reach a point in the timeline where the end-user adoption begins to increase at a much faster pace. At this point the majority of the market value is going to shift away from technology development and more towards system integration and service support for the end-user. While Dust Networks has some experience with system integration and support, at their current size they cannot realistically provide this level of service on a large scale. Instead they need to ensure the future adoption of this technology, by focusing on developing the software and creating innovative ways in which the technology can be used. Once the adoption of the technology begins to accelerate, they can begin to look at increasing the size and scope of their company to include end-user integration and software support services.



Opportunity Recognition (Module 1)

Within the WSN market, there has not emerged a killer application that would lead to widespread adoption of the technology. Dust Networks is focusing on three major markets; building automation, industrial monitoring, and security. While there has been some headway into these markets, overall growth has been relatively slow. In order for Dust Networks to continue to succeed in increasing adoption of their technology they must position their company in ways that enables them to recognize new opportunities and quickly move to take advantage of these opportunities.

To do this, Dust Networks should look to the examples set by Honda as they expanded into the US motorcycle market. When Honda initially entered the market they targeted the existing large motorcycle market. After realizing slow sales and design problems for their motorcycles in this market, they recognized an opportunity for their smaller lightweight *Supercub*. By being able to quickly adapt and switch gears to manufacture and sell this new bike, they were able to have phenomenal sales and redefine the US motorcycle industry. Another example of how Honda was able to adapt was the introduction of the “Nicest People” advertising campaign. By breaking away from their traditional advertising and launching this new message, Honda was able to target an untapped segment of the marketplace, further increasing the sales of their *Supercub*.

According to Richard T. Pascale, one of the principle reasons behind Honda’s success is that they showed the ability “to experiment, to learn quickly from mistakes, to rapidly revise design problems, and to discover new opportunities.”[1] Pascale define organizational agility as speed and adaptiveness and theorizes that it is a core competence that must be integrated into all aspects of a company’s culture and corporate strategy.

[1] Source: CMR Forum: The “Honda Effect” Revisited. CMR, Volume 38, Number 4, Summer 1996

Dust Networks is going to have to rely on this type of organizational agility if it wants to ensure its success going into the future. While Dust Networks does not have the advantage of years of experience that Honda did, they do have an advantage in that they are a small company. The small size of Dust Networks will allow them to react quickly to new opportunities and not be slowed down by the bureaucracy that is often found in large organizations. Also, by making sure that opportunity recognition is an integrated component of the company and not just an independent process, they will be better equipped to recognize opportunities in the future when the company becomes larger.

A real-world chance for Dust Networks to test this theory is through a recent partnership they entered with the U.S. Department of Energy (D.O.E.). One of the first stages of this partnership is going involve creating a wireless automatic lighting control system that will be used in buildings to reduce power consumption. Throughout the duration of this partnership Dust Networks will no doubt be developing new software and technology that will have the potential to be adapted for other uses. By creating a company culture around organizational agility, Dust Networks will be able to benefit from these opportunities and increase the adoption of wireless sensor networks.

Team Structure (Module 3)

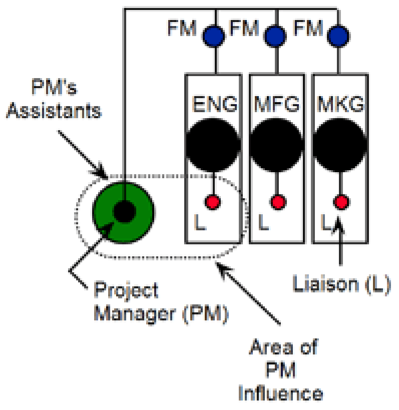
In addition to creating a culture centered on recognizing and reacting to opportunities, Dust Networks needs to have a team structure that most effectively manages their technology. According to research conducted by Clark and Wheelwright, there are four different types of team structures; functional teams, lightweight teams, heavyweight teams, and autonomous (or tiger) teams. Functional teams group people primarily by their particular discipline, each below a different functional manager. A lightweight team is similar to a functional team with the addition of a project manager that coordinates all the activities of the individual functional groups. The heavyweight team is often considered the most optimal type of team and it incorporates a functional leader that has direct responsibility and control over a core team of individuals in each of the different disciplines. The forth team is unique in that it does not need to follow any of the guidelines of the existing organization. This tiger team is able to make its own rules and is completely responsible for its success or failure.

When Dust Networks was established in 2002, the structure of the company was that of a tiger team. With only four employees, it was the only team structure that made sense. Each of the members of the team had to be extremely focused on making sure that the company stayed afloat. It required a lot of coordination and cross-functional integration. As Dust Networks grows, the type of team structure will have to evolve to successfully manage its development process.

Currently Dust Networks has grown from the four founders to between 20 and 50 employees. (Kris Pister, one of the founders and the person we met with a couple times did not feel comfortable discussing the size of the company with us. Between 20 and 50 is an estimate based on the last time Sukun visited the company and the fact that they recently had to move to a larger location) Because of this growth, a tiger team is no longer a feasible organizational structure. While this type of team has the advantage of being results oriented and focused, it also has the disadvantage of having weak integration with the rest of the organization. With a maximum size of 50 employees, Dust Networks can not afford to have poor integration with any member of the organization, let alone an entire team.

The small size of Dust Networks also makes it impractical for them to be structured into heavyweight teams. With between 20 to 50 employees, we can assume that they have about 10 employees in research and development. With three product lines, these 10 employees would have to be sub-divided into three different R&D groups. Therefore, despite the many advantages of a heavyweight team, until the company grows larger, heavyweight team structures will not be a viable option.

Looking at the last two team structures, since a lightweight team is a slight improvement over the functional team, we recommend that Dust Networks structure their company into lightweight teams to manage their product lines. Like the functional team, the lightweight team has the advantages of alignment of responsibility and authority, the managers who evaluate employees' work are the same managers that make decisions about career paths, and this structure utilizes functional expertise. In addition, the project managers on lightweight teams improve communication and coordination. Despite these improvements, lightweight teams often fail to realize substantial improvements in speed, efficiency, and project quality. This lack of results is a consequence of the power still residing with the functional managers and not with the project manager. To prevent this from happening, the executive team should look for ways to give more decision-making power to the project managers. Perhaps they can find a middle ground between a lightweight and heavyweight team where project managers have authority, but where individuals do not have to be divided into sub-groups for each of the different product lines.



As Dust Networks continues to grow, they will need to determine whether or not to incorporate additional types of team structures. For development, heavyweight teams may work best. However, if the company begins to move more towards a service model, a lightweight team may be best for training new service employees. In the meantime, Dust Networks must make sure that they create a company culture that will be conducive to including any of the four team structures.

Hardware and Protocol Standards (Module 2)

Wireless sensor networks are in their early stages of development, and standards are not well established. Some companies come up with their own internal standards. Since architecting WSN systems is not very difficult, start-up companies could architect their system by themselves. However, there are companies waiting for standards to emerge. An example of this is in the PC market. Dell focuses on standards-based technology and they enter the market at the commodity phase when all standardization is finished. This reduces the risk coming from an uncertain future of competing standards. In WSN, Intel is an example of a company waiting until the market enters a standardization phase. Intel is component manufacturer who wants to produce both the microcontroller and the radio chip for WSN. Intel is spending a huge amount of R&D (actually only R) budget in WSN, but all the money goes to supporting TinyOS, and developing pilot application project using WSN. This investment acts to grow the market as quickly as possible to a threshold where Intel can justify entering WSN market. The investment also allows Intel to increase its influence on the WSN research and finally on the standardization. After standards are formed in favor of Intel, then Intel can manufacture products without uncertainty or risk. If influential component manufacturers like Intel control platform architecture and standards, they may drain value from network manufactures like Dust Networks.

<u>The Need for Standards from System Integrators</u>

Within the WSN market system integrators are longing for standards. System integrators want to be able to purchase networks from different network manufacturers, and to mix them.

For example, system integrators want to buy motes from Dust Networks, and a network manger from Ember, and make them talk to each other. And beyond substitution, compatibility is also a compelling reason for standards in WSN. System integrators need to integrate existing systems built using networks from different venders. More severe cases happen when one vendor goes out of business. Without compatibility, all the networks built using that company's product need be completely replaced by products from another company.

Strong needs for standardization from system integrators is a good chance for Dust Networks. Since Dust Networks is a late starter compared to Crossbow and Ember, Dust Networks has more advantage in entering the competitors' markets through standardization. If Dust Networks' products do not follow standards properly, system integrators choose standard products of other vendors.

Type of Standards in WSN

In WSN, standards are primarily centered on communication. Interface to sensor boards is one example. Physical connector shape, layout and usage of pins can be standardized, so that one sensor board can be used by multiple network manufacturers. Like IEEE 802.11 WLAN for the Internet, high-level communications is becoming a standard for Radio chips in the WSN market..

WSN Hardware and Protocol Standards

Hardware-level radio chip standard is set in 2004. IEEE 802.15.4 PAN (personal area networks) states physical behavior of radio communication. Crossbow and Ember were quick at adopting this radio standard. Dust Networks was more than six months behind these competitors in launching a 802.15.4 compliant product. Communication protocol Zigbee is emerging. This standard will enable motes from different vendor to communicate, even though they may not understand the meaning of data. (Only if the application layer is known can motes from different manufacturers understand each other.) Ember is very active in setting Zigbee standard, and Crossbow is also a member of Zigbee alliance. However, Dust Networks is still not a Zigbee member. In these areas, Dust Networks is behind competitors in standardization.

Strategy for Dust Networks in Standardization

System integrators show a strong need for standardization and component manufacturers like Intel are trying to increase their control over platform architecture and standardization. As a result, network manufacturers will have weak appropriateness in the future. Dust Networks doesn’t have a lion’s share in WSN like Dell’s in the PC market. Dell could achieve virtual innovation by leveraging its market share, and benefit from removed uncertainty on competing standards. Dust Networks can’t do the same thing. However, it can adopt confirmed standards quickly, and participate in standardization to not fall behind competitors. And partnership with influential component manufacturer can reduce current disadvantage, and increase its ability for virtual innovation.

Open versus Closed Software (Module 4)

Open Software

Open software is software which is freely distributed with source code, and whose source code can be modified and contributed by anyone. Since anyone can contribute, there can be millions of developers which a private company can not afford. There are also more users who report bugs. Developers are free from time and budget, so they can develop software in a way they think right. Developers actually use the software, and they make it for diverse reasons: one’s own use, acknowledgement from community, etc. This creates a situation where cutting-edge technology is continually discovered and tested, which is ideal from a scientific view.

Linux, an operating system, is one of the most famous examples of open source software. Red Hat distributes its own stable version of Linux and creates value by providing stable versions and customer support.

Closed Software

Closed (or proprietary) software is software which is distributed in binary executable format only. Some of them are free to use, and some of them require purchase or license fees. Microsoft Windows is an example of closed software. Usually proprietary software is developed by a company for commercial purpose. Closed groups of people develop it under the control of a company. Developers are not necessarily users, and they participate in development for profit. Company searches for user needs, designs, implements, and tests the software. Since the software is not released and tested by users during development, there usually exists a large team for internal testing. Often closed software is distributed to a small number of lead users before a major release; this is called a beta test.

Open versus Closed

Open software can work only when a developer is a end-user. Even though it is ideal for general purpose operating systems, it will not be appropriate for specific applications like dental office billing software. Since open software evolves continuously, users will benefit from faster software improvements. However, it may not guarantee backward compatibility. Since there are lots of users in open software, it is tested extensively and debugged. However, as the software evolves new bugs are introduced and no one is responsible for bugs in open software. Critical software applications can not rely on this approach.

The Difference between PC software and WSN software

PC software is made for use by human who directly interact with it. There are many PC software programs used in everyday life. For example, operation systems, web browsers and media players are widely used by many diverse users. They fulfill general and diverse needs of human directly. So people use it, feel it, and find problems.

In contrast, WSN software, which is an example of embedded software, interacts with environments and machines. The software performs very specific tasks and the human needs are performed by the machines it controls indirectly.

Examples in WSN

Dust Networks and Ember uses their own closed software solutions. Crossbow uses an open software call TinyOS. TinyOS is developed in UC Berkeley for WSN research. Crossbow has a 3-party partnership with Intel and UC Berkeley. Intel wants to increase market size as quickly as possible. So Intel supports research at UC Berkeley, and support Crossbow as a rabbit (pioneer lead user of technology) by providing financial investment. Crossbow collaborates with UC Berkeley and exchanges technical assistance. As a result, Crossbow uses the TinyOS of UC Berkeley and receives diverse technical assistance.

Open versus Closed in WSN

Since embedded software fulfills specific needs of machines, unlike Linux, developers are not direct users. The usage of software is so narrow and specific for each application, it is not often transferable to other cases. Hence, software is not widely reused by other members of community. Even if it is reused by other users, individuals usually don’t have close contact to the software. So it is hard to find bugs in software.

In WSN, the cost of improvements is not free. We can not simply download and install new components, like you can with new versions of Red Hat Linux software. In WSN, newly downloaded software has to be manually installed one each physical motes. It requires significant amount of labor.

Robustness of software is very important in WSN. Many typical uses of WSN involve critical missions. For example, when they are used to control machines, malfunction of motes can lead to disastrous operations of machines. In the case where they are reporting hazards, false positives from malfunctioning software can be very costly. Moreover, since machines don't have intelligence, malfunction of software can not be handled in real-time. When the Netscape web browser does not function properly, a person can restart the browser. However, machines can't handle wrong command packet. Open software doesn't provide a guarantee for robustness. When the entire software is tightly controlled by one entity, guaranteeing quality control is easier through redundancy or watchdog mechanisms.

In-house versus University Resources

Currently open software, TinyOS, is supported mainly by graduate students and researchers. When the technology cycle progresses and WSN becomes less interesting as a research topic, they may switch to other area. Then, open software will not be supported any more, and companies like Crossbow will lose their advantage. In addition to that, in exchange for technical support from UC Berkeley and financial investment from Intel, Crossbow publishes all information about products and software. UC Berkeley wants to increase impact of its research, and Intel wants to increase market size quickly. As a result, clones of Crossbow's product are emerging in Europe and Asia. For example, there are already three companies in Korea producing clones compatible with Crossbow's products and they are quickly expanding market share in Korea. When they begin exporting to the US market, Crossbow will face severe competition within its domestic market. In contrast, if Dust Networks develops software and accumulate technology in house, it can keep improving software and provide a unique solution.

All these points, especially the deviation of developers and users, make closed software model very compelling for the embedded software in WSN.

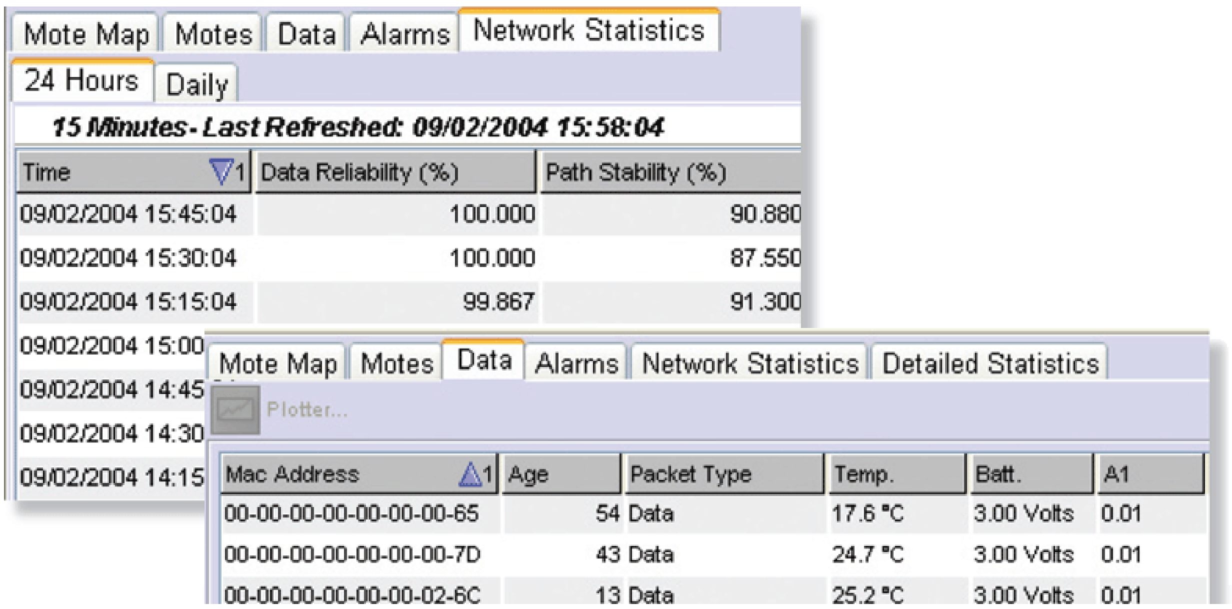
Conclusion

Moving into the future, wireless sensor networks are going to provide tremendous opportunities for all players within the market. While Dust Networks is not the largest company in the space, they have the potential to realize substantial future growth and success. The magnitude of this success is going to be largely based on their ability to move into the services side of this industry once adoption of this technology begins to take-off. In the meantime, they will need to make sure that they create a company culture that is centered on opportunity recognition and build teams that will have the speed and agility to capitalize on these opportunities. In addition to these organizational changes, they will need to continue to adopt hardware protocol standards while maintaining the value of their WSN through proprietary network software. By following this strategy, Dust Networks will be in the best position to thrive in the WSN market.

Exhibit 1 – History of Dust Networks

July 2002 -	Company Founded by Kris Pister and Rob Conant
January 2003 -	Pister takes leave from UC Berkeley to work at Dust Networks full-time
February 2002 -	Series A funding of $7Million
August 2004 -	First SmartMesh product officially shipped
December 2004 -	Kris Pister returns to UC Berkeley & consults at Dust Networks once a week
January 2005 -	Series B funding of $22Million
February 2005 -	Partnership with U.S. Department of Energy

Exhibit 2 – Screen Shots of SmartMesh Software

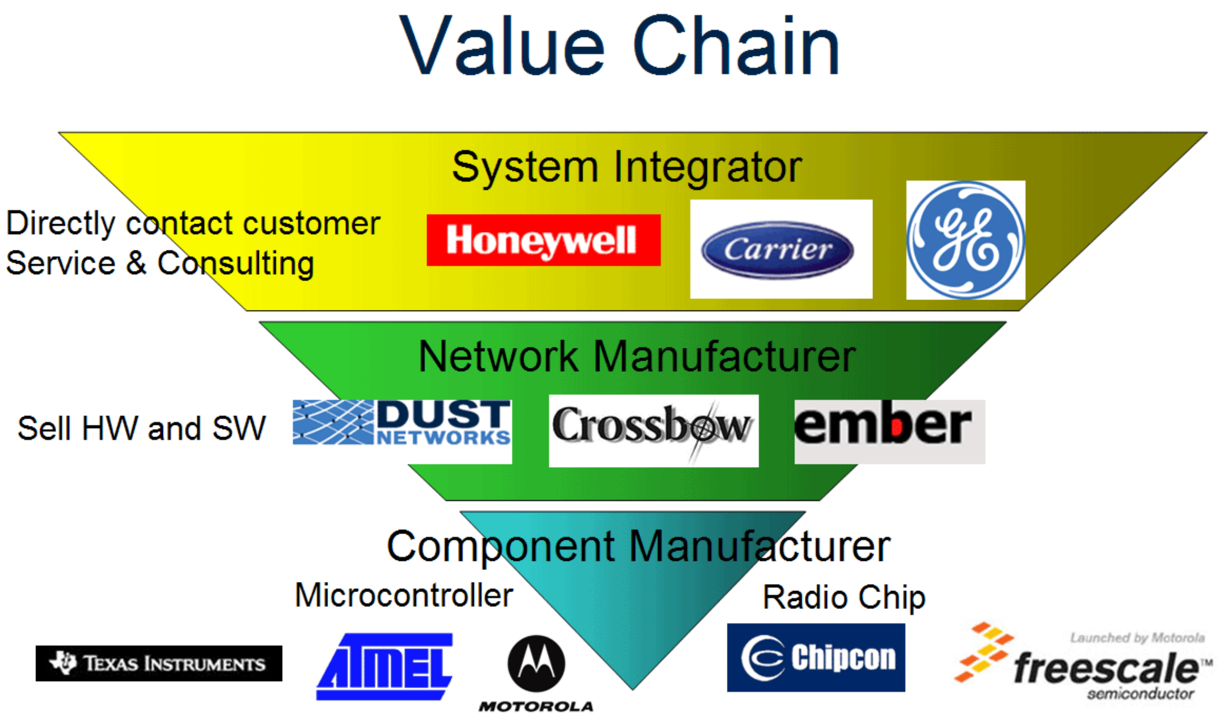


15 Minutes - Last Refreshed: 09/02/2004 15:58:04

Time	Data Reliability (%)	Path Stability (%)
09/02/2004 15:45:04	100.000	90.880
09/02/2004 15:30:04	100.000	87.550
09/02/2004 15:15:04	99.867	91.300
09/02/2004 15:00		
09/02/2004 14:45		
09/02/2004 14:30		
09/02/2004 14:15		

Mac Address	Age	Packet Type	Temp.	Batt.	A1
00-00-00-00-00-00-00-65	54	Data	17.6 °C	3.00 Volts	0.01
00-00-00-00-00-00-00-7D	43	Data	24.7 °C	3.00 Volts	0.01
00-00-00-00-00-00-02-6C	13	Data	25.2 °C	3.00 Volts	0.01

Exhibit 3 – WSN Value Chain


Value Chain
System Integrator
Directly contact customer
Service & Consulting
Honeywell
Carrier
GE
Network Manufacturer
Sell HW and SW
DUST NETWORKS
Crossbow
ember
Component Manufacturer
Microcontroller
Radio Chip
Texas Instruments
Atmel
Motorola
Chipcon
Launched by Motorola
freescale semiconductor