Towards a Sensor Network Architecture: Lowering the Waistline

David Culler[*], Prabal Dutta[*], Cheng Tien Ee[*], Rodrigo Fonseca[*],
Jonathan Hui[*], Philip Levis[*], Joseph Polastre[*], Scott Shenker[*†],
Ion Stoica[*], Gilman Tolle[*], and Jerry Zhao[†]

1 Introduction

Wireless sensor networks have the potential to be tremendously beneficial to society. Embedded sensing will enable new scientific exploration, lead to better engineering, improve productivity, and enhance security. Research in sensor networks has made dramatic progress in the past decade, bringing these possibilities closer to reality. Hardware, particularly radio technology, is improving rapidly, leading to cheaper, faster, smaller, and longer-lasting nodes. Many systems challenges, such as robust multihop routing, effective power management, precise time synchronization, and efficient in-network query processing, have stable and compelling solutions. Several complete applications have been deployed that demonstrate all of these research accomplishments integrated into a coherent system, including some at relatively large scale [5, 17].

But the situation in sensornets, while promising, also has problems. The literature presents an alphabet soup of protocols and subsystems that make widely differing assumptions about the rest of the system and how its parts should interact. The extent to which these parts can be combined to build usable systems is quite limited. In order to produce running systems, research groups have produced vertically integrated designs in which their own set of components are specifically designed to work together, but are unable to interoperate with the work of others. This inherent incompatibility greatly reduces the synergy possible between research efforts and impedes progress.

It is the central tenet of this paper that the primary factor currently limiting research progress in sensornets today is not any specific technical challenge (though many remain, and deserve much further study) but is instead *the lack of an overall sensor network architecture*. Such an architecture would identify the essential services and their conceptual relationships. Such a decomposition would make it possible to compose components in a manner that promotes interoperability, transcends generations of technology, and allows innovation.

2 The Nature of an Architecture

At the highest level, an architecture decomposes a problem domain into a set of *services*, which are functional components, their mechanisms and their responsibilities. An architecture can also define a set of *interfaces* to its services, which are the structures and functions services expose their mechanisms with. Finally, at the lowest level, an architecture can specify its *protocols*, which include packet formats, communication exchanges, and state machines.

For interfaces and protocols, we say an architecture *can* define them because sometimes it is advantageous not to. For example, the Internet architecture precisely defines IP as a service (end-to-end communication, best-effort delivery, fragmentation/defragmentation, etc.) and as a protocol (packet format and semantics), but is ambivalent to the IP interface. Given the Internet's principal design goal — it is a *network* interconnection architecture — this ambivalence makes sense: IP does not want to dictate what software runs on each host.

In contrast to the Internet architecture, which seeks to promote communication interoperability, the POSIX architecture cares about software interoperability. Correspondingly, it cares greatly about interfaces while remaining ambivalent about the protocols. For example, the sockets service for end-to-end communication provides a precise interface, but has several underlying protocols (e.g., local communication, TCP, etc.).

The challenge in sensor networks is that their modes of operation introduce requirements and tradeoffs are very different from traditional systems. A sensor network application dictates sensor modalities, sample rates, real-time processing, data storage, and information exchange protocols among nodes. Early vision papers and analyses claimed that the traditional application/OS and network/data-link divisions are not well suited to sensor networks [3, 7], and community experiences building protocols and applications have shown this claim to be true [11, 16, 18, 27]. Thus, current sensor network software systems, such as TinyOS [9] relax these divisions and give developers the flexibility to define new ones. This relaxation has allowed researchers to re-examine core issues in scheduling, power-control,

[*] CS Division, UC Berkeley, Berkeley, CA 94720.
[†] ICSI, 1947 Center Street, Berkeley, CA 94704

and information flow by cutting across traditional service boundaries [14].

Cutting across boundaries, however, has led to monolithic solutions or to subsystem components with arbitrary interface assumptions. While research groups have each been able to build large and complex systems, the resulting services, interfaces, and protocols are incompatible with each other. For future work to be able to build on the prior efforts of others, we need a sensor network architecture, which will re-establish a meaningful separation of concerns.

The Internet architecture demonstrated how a properly chosen set of guiding principles and services can shape the evolution of a complex system over vast changes in technology, scale, and usage [2]. The philosophy of designing for heterogeneity, change and uncertainty was a radical shift from classical systems design, which more traditionally seeks a near optimal assembly of near optimal parts. Faced with integrating several existing networks with widely varying characteristics, the end-to-end principle and focus on interoperability led to a design that has successfully coped with tremendous growth and change. However, this design is not free of costs; the use of rigid layering sacrifices efficiency in various regards in return for increased interoperability.

The power of the Internet is revealed not so much in the elegance or efficiency of its individual services, but in its overall ability to adapt. This is one of our goals for developing an architecture for sensornets. We must be extremely mindful of any loss of efficiency for particular tasks as we seek to greatly enhance the interoperability between components and ability to advance.

The experiences and efforts of the sensor network community over the past years has helped discover exactly how the requirements and concerns of a sensor network architecture are different from the Internet, and how they are the same. The challenge in defining a sensor network architecture is deciding what to specify in its services and what to leave open. Specifying too little will force systems to re-implement functionality they cannot depend on, while specifying too much will constrain future technologies and possibly lead developers to discard the architecture. For this reason, we expect developing an architecture to be at first a growing and organic process. While conclusions from community experience have clearly converged on some issues, such as packet timestamps, others, such as aggregation, are still under debate. By starting with services (or even parts of services) for which there is consensus, an architecture will help focus the research debate on open problems, promoting forward progress.

3 The Narrow Waist

A complete sensornet architecture will need to address a family of specific issues, such as discovery, topology management, naming, routing and so on, but the overriding question is whether there is a “narrow waist” — a functional component representing a common service that permits a wide variety of uses above and a range of implementations below. At what level should it occur and what should it express? By requiring all network technologies to support IP, and all applications to run on top of IP, the Internet accommodates, even encourages, a vast degree of heterogeneity and diversity in both applications and underlying technologies. We have an analogous goal for sensornets; in both the application and device arenas we are in the midst of extremely rapid developments. Sensornets will only flourish if we can identify a narrow waist in the architecture that will allow devices and protocols to evolve and change without hampering optimization. The Internet has shown that the most important service of a network architecture is its narrow waist.

We claim that sensor networks can also have a narrow waist, the Sensor-net Protocol (SP). Unlike IP, which is a multihop protocol intended for end hosts communicating over a shared routing infrastructure, SP is a single hop protocol. The reason for this difference is simple: sensor networks use a wide range of multihop protocols, such as dissemination [15], flooding, tree routing [26], and aggregation [18]. Applications differ dramatically in their communication patterns and are intimately tied to their associated network protocols. Most applications neither require nor benefit from a common, universally routable addressing scheme. Those that do can build such protocols on top of SP.

The first step in developing our architecture is defining the SP service by deciding which mechanisms and functionality it provides and which it does not. Using SP, protocol designers must be able to design a range of efficient routing protocols independently of the underlying link layer. SP must facilitate in-network processing and collective communication as well as point-to-point transport. Moving the point of universal abstraction downward presents new issues that we do not typically concern ourselves about in the Internet architecture. It also requires a careful design of the layers above SP to provide a reasonably general platform on which to build various sensor network applications efficiently. If SP is to be a well defined service on top of a range of physical layers, how functionality divides across the packet boundary is a key question. To support the network protocols found in the sensornet literature, the mechanisms which a sender should be able to control include link level acknowledgments, post-media arbitration timestamping, retransmission and power manage-

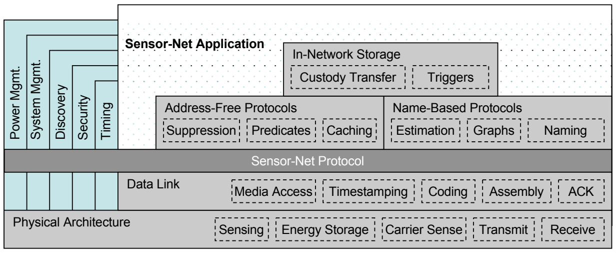

Figure 1: Sensor Network Service Decomposition

ment (cf. [20]). In addition to providing control points downwards, SP needs to expose costs upwards to higher layers so protocols can receive feedback on how to optimize their behavior as they exercise the available control.

For example, it is clear from community experience that the SP service must provide packet timestamps. Time synchronization research [6, 13] has shown that obtaining high precision timestamps on packet transmission and reception is inexpensive and can enable a wide range of synchronization algorithms above. While the need for this information is clear, exactly how it manifests is less so. There is consensus that when a node receives a packet, the SP service must provide a receive timestamp. As this timestamp is not a field of the packet that is received over the air, it is part of the SP service but not the protocol. The point of debate is on transmission. ETA [13] argues that transmitted packets should contain the sender's timestamp, while RBS [6] argues that this is unnecessary, as only the transmitter needs to know its timestamp. While both agree timestamps must be part of the SP service, ETA requires transmit timestamps to be part of the SP protocol, while RBS does not.

SP sits below many multihop protocols. Allowing higher level protocols to share control over an underlying communication medium raises concern as to how these protocols work together and cooperate. This is just the kind of investigation that the existence of SP would promote. We suggest that this question is tractable and very interesting in sensornets because they typically host a small number of widely distributed applications. In the Internet, such control is problematic because the infrastructure is shared by arbitrary applications anywhere in the world. The application specific nature of sensornets is more conducive to cross-layer and cross-application customization.

Therefore, rather than immediately specify protocols, our development of a sensor network architecture starts with defining SP as a service and providing a possible interface to that service so developers can test and evaluate it. Once, through literature analysis, communication with the community, and, of course, trial and error, we determine the boundaries of SP as a service, we can then focus on building and evaluating different candidate SP interfaces and SP protocols. We have begun this process by making a first attempt at defining SP as a service [21]. Trying to define a common service on top of very different underlying link layers (e.g., TDMA and CSMA) raises interesting questions about networking in this regime and suggests places where well-established networking terminology is ill-suited.

4 Filling In the Architecture

SP is the keystone of our sensor network architecture, bridging higher level protocols and applications to underlying data link and physical layers. Defining the SP service requires understanding the requirements of applications that lie above it and the capabilities of the technologies that lie below. Just as with IP, it is unlikely that SP will be ideally suited to all of its possible uses. However, by examining applications and their requirements, we can make educated decisions on what tradeoffs SP makes between its above and below pressures.

Applications today use a wide range of service layers, some of which have no clear analogues in the OSI model. For example, several commonly used communication services, such as collection routing [26] and dissemination [15], are *address-free*, in that, from the perspective from an application, there is no explicit destination. Of course, there are also name-based communication services, but the form and semantics of the naming are very different than end-to-end communication.

Address-free and name-based communication represent traditional service layers, which encapsulate underlying functionality. Our sensor network architecture also has *cross-layer services*, which cut across SP and indeed the entire architecture. Deployed applications have demonstrated that there are pieces of information and functionality which many different services require concurrently. Establishing the concept of cross-layer services allows existing approaches to continue while providing the structure necessary to promote composability and reuse.

4.1 Proposed Decomposition

Figure 1 shows a possible decomposition of a sensor network architecture. SP is the unifying service that bridges protocols and applications to the underlying data link and physical layers. Situated above SP are multiple network layer services, with applications selecting specific ones that suite the networking needs of the application. In Section 4.2, we discuss two of these upper layer services, name-based and address-free protocols. Situated below SP are underlying data link and physical layers, such as 802.15.4 [25] or S-MAC [28]. The diversity of functionality underlying layers present poses a variety of technical challenges to SP's design, which we discuss and address in our SP proposal [21].

In addition to the layered services above and below SP, the architecture has cross-layer services, which Figure 1 shows on the left side. Cross-layer services include power management, timestamping/time synchronization, and discovery. As we discuss further in Section 4.3, these services are cross-layer in that they have uses across the entire spectrum of service layers.

This decomposition is far from complete. As sensor networks evolve and spread into new application domains, it is inevitable that new services will emerge. Current and foreseen future uses motivate our current decomposition, but it is also intended to be flexible enough to engender growth.

4.2 Address-free and Name-based

Unlike an IP network, which supports a single network addressing scheme and largely provides a single communication abstraction (*i.e.*, unicast), applications developed so far use a variety of naming schemes and multihop communication services. For example, the Line in the Sand event detection application routed along a 2D grid [5], the Great Duck Island habitat monitoring application routed up a collection tree [17] and Pursuer Evader Game tracking application used a landmark routing overlay on top of a tree-building algorithm [23].

This variety in naming and multihop communication is one of the main reasons behind our decision to push the narrow waist below the OSI network layer. Lowering the narrow waist allows the architecture to express and encompass this diversity both in the present and in the future. Trading off between the requirements of higher level services and the desire to keep SP as simple as possible is the principal first challenge in developing the architecture. In the remainder of this section, we describe two higher level services and how they might influence SP. Key architectural issues that arise in designing these services include route discovery and maintenance, naming, and the packet forwarding rules.

The address-free service layer encompasses a wide range of protocols, including flooding, collection routing [26], dissemination [15], and aggregation [4]. Although these protocols may include names to refer to data items — such as sequence numbers or dispatch IDs — they do not identify nodes directly. For example, when an application wants to send a piece of data up a collection tree, it does not need to specify a destination because it is implicit: the node's parent in the tree. The underlying collection tree routing protocol may address the parent directly, but it encapsulates this naming and hides it from layers above.

Unlike collection routing, however, which typically names nodes at the SP level, broadcast and dissemination protocols rely on the implicit naming provided by local connectivity. This represents an interesting SP design consideration, as some underlying MAC layers (e.g., TDMA-based MACs that turn off the radio) may not by themselves provide an efficient local broadcast primitive. This tension between the requirements of layers above and the capabilities of layers below demonstrates some of the difficulties that designing SP presents.

The name-based service layer encompasses multihop communication based on destination identifiers. This includes approaches such as geographic routing [12] and logical coordinate routing [8, 19], as well as more abstract and flexible naming schemes such as directed diffusion, which use data identifiers [11]. Global network names are powerful enough to support content-based storage within the sensor network, but require any-to-any routing [22].

In addition to packet forwarding, a node along a path can inspect received data and make local decisions regarding a packet based on its contents, possibly transforming the data before forwarding it, or suppressing it completely. This in-network processing can reduce communication while keeping higher-level semantic requirements. For example, when collecting a MAX query, which returns the maximum value of some variable, nodes need only forward the highest value they receive and suppress all other values.

The key observation is that the services above SP support very different semantics than those found in the network layer services of the Internet and OSI specifications. In particular, sensornets are primarily concerned with dissemination, collection, aggregation, and gradient-directed services, whereas the Internet is principally concerned with end-to-end communication [1].

4.3 Cross-Layer Services

One novel aspect of our sensor network architecture is the concept of cross-layer services. These services cut across layers or arise within multiple layers. Instead of being fully encapsulated at one layer, only visible to the layers above and below, cross-layer services are accessible to all of the layers in the system. In this section, we use power management to motivate the need for cross-layer services in a sensor network architecture, describe some of the research challenges they pose, and present timestamping as one example of such a service.

Energy constraints are a defining characteristic of sensor networks. Traditionally, power aware networking has dealt with a single point in the stack in isolation. This approach is not practical in sensornets because power management often appears in many places and takes many forms. Below SP, power aware MACs attempt to turn off the radio invisibly to the stack above [24]. Within SP, buffering multiple packets and sending them back-to-back in a burst can be more efficient than sending them individually as they appear from the network layer services [21]. Routing layers above SP can have multihop flow information that allows them to schedule future radio activity [10]. Applications can have their own scheduling policies: TinyDB shuts down the whole networking stack between query processing epochs [16].

As a consequence of its ubiquity, power management is particularly challenging to abstract into a clean architectural concept. The architecture must allow many different services from very different levels to collaborate and work together. These services must therefore be accessible to all levels of the system. On one hand, these services must have policies to arbitrate between conflicting requests; on the other, constraining the possible policies unnecessarily will hamper future growth. An architecture must establish clear guiding principles and sufficiently rich, yet loosely-coupled and appropriately abstracted, interfaces to support cross layer services.

All of the power management approaches mentioned above use some form of time synchronization to schedule communication. All of these time synchronization algorithms depend on having accurate packet timestamps. Therefore, these timestamps must be information that cuts through layers so sub-SP as well as super-SP services can use them. While time synchronization services can be situated above SP, MAC-layer timestamping below SP greatly improves their precision [13]. By choosing an to generate this data at an idealized point in the communication stack (e.g., post media arbitration) SP can achieve microsecond resolution inexpensively.

Timing information must cross layers so many services can take advantage of them. The sensor network architecture therefore provides it in cross-layer services. The preferred method of exposing timestamps in the link interface, and more generally across the architecture, is an important design point that must be addressed with an eye toward removing any temptation for time coordination services to circumvent SP. Power management is an example of a cross-layer service for downward control; timestamps are an example of a cross-layer service for upward information flow.

While this section presented only two cross layer abstractions, there are many more that we need to address. Examples include system management, discovery, and security. These services need to be accessible to all of the layers in the system so their abstractions present a central challenge to the architecture's design: developing a methodology for providing interfaces rich enough for application/system collaboration while remaining flexible enough to encompass growth and evolve as time rolls forward.

5 Conclusion

We contend that the main obstacle limiting progress in sensornet work is the lack of an architecture. A sensor network architecture would factor out the key services required by applications and compose them in a coherent structure, while allowing innovative technologies and applications to evolve independently. We argue that the narrow waist of this architecture should not be a network layer as in the current Internet, but single-hop communication with a rich enough interface to allow a diverse range of network protocols. This design decision is driven by the fact that, unlike an IP network, sensornets require a wide variety of naming schemes and multihop communication services.

However, there are many questions that need to be answered before such an architecture becomes a reality. Chief among those are the functionality provided by the SP service, the functional decomposition of sensor networking into services now that the narrow waist is single hop, and how cross-layers services such as timestamping can be designed to enable a broad spectrum of uses while minimizing complexity. Our hope and goal is that such an architecture will enable research groups to more easily collaborate and build on each other's efforts. Rather than a set of incompatible and vertically integrated systems, we will in the near future see in sensor networks the variety and innovation we see in the Internet today.

Acknowledgements

This work was supported by the National Science Foundation, under grant CNS-043545.

References

[1] B. Carpenter. Architectural principles of the internet. Request For Comments 1958, June 1996.

[2] D. D. Clark. The design philosophy of the DARPA internet protocols. In *Proceedings of the SIGCOMM '88 Conference*.

[3] N. R. C. Committee on Networked Systems of Embedded Computers. *Embedded, Everywhere: A Research Agenda for Networked Systems of Embedded Computers*. National Academy Press, Washington, DC, USA, 2001.

[4] J. Considine, F. Li, G. Kollios, and J. Byers. Approximate aggregation techniques for sensor databases. In *Proceedings of the 20th International Conference on Data Engineering (ICDE '04)*.

[5] P. Dutta, M. Grimmer, A. Arora, S. Bibyk, and D. Culler. Design of a wireless sensor network platform for detecting rare, random, and ephemeral events. In *Proceedings of the Fourth International Conference on Information Processing in Sensor Networks (IPSN '05)*.

[6] J. Elson, L. Girod, and D. Estrin. Fine-grained network time synchronization using reference broadcasts. In *Proceedings Fifth Symposium on Operating Systems Design and Implementation (OSDI 2002)*.

[7] D. Estrin, R. Govindan, J. S. Heidemann, and S. Kumar. Next century challenges: Scalable coordination in sensor networks. In *Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom '99)*.

[8] R. Fonseca, S. Ratnasamy, J. Zhao, , C. T. Ee, D. Culler, S. Shenker, and I. Stoica. Beacon vector routing: Scalable point-to-point routing in wireless sensornets. In *Proceedings of the Second USENIX/ACM Symposium on Network Systems Design and Implementation (NSDI 2005)*.

[9] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. E. Culler, and K. S. J. Pister. System architecture directions for networked sensors. In *Architectural Support for Programming Languages and Operating Systems*, pages 93–104, 2000. TinyOS is available at http://webs.cs.berkeley.edu.

[10] B. Hohlt, L. Doherty, and E. Brewer. Flexible power scheduling for sensor networks. In *Proceedings of the Third International Symposium on Information Processing in Sensor Networks*, Berkeley, CA, Apr. 2004.

[11] C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed diffusion: a scalable and robust communication paradigm for sensor networks. In *Proceedings of the Sixth Annual International Conference on Mobile Computing and Networking (MobiCom '00)*.

[12] B. Karp and H. T. Kung. GPSR: greedy perimeter stateless routing for wireless networks. In *Proceedings of the Sixth International Conference on Mobile Computing and Networking (MobiCom 2000)*.

[13] B. Kusy, P. Dutta, P. Levis, M. Maróti, Ákos Lédeczi, and D. Culler. Elapsed time on arrival: A simple, versatile, and scalable primitive for time synchronization services. *International Journal of Ad hoc and Ubiquitious Computing*, 2005.

[14] P. Levis, S. Madden, D. Gay, J. Polastre, R. Szewczyk, A. Woo, E. Brewer, and D. Culler. The emergence of networking abstractions and techniques in tinyos. In *Proceedings of the First USENIX/ACM Symposium on Network Systems Design and Implementation (NSDI)*, 2004.

[15] P. Levis, N. Patel, D. Culler, and S. Shenker. Trickle: A self-regulating algorithm for code maintenance and propagation in wireless sensor networks. In *Proceedings of the First USENIX/ACM Symposium on Network Systems Design and Implementation (NSDI)*, 2004.

[16] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. TAG: a Tiny AGgregation Service for Ad-Hoc Sensor Networks. In *Proceedings of the Fifth ACM Symposium on Operating System Design and Implementation (OSDI 2002)*.

[17] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson. Wireless sensor networks for habitat monitoring. In *Proceedings of the ACM International Workshop on Wireless Sensor Networks and Applications*, 2002.

[18] S. Nath, P. B. Gibbons, Z. Anderson, and S. n Seshan. Synopsis diffusion for robust aggregation in sensor networks. In *Proceedings of the Second ACM Conference on Embedded Networked Se nsor Systems (SenSys)*, 2004.

[19] J. Newsome and D. Song. Gem: graph embedding for routing and data-centric storage in sensor networks without geographic information. In *Proceedings of the first international conference on Embedded networked sensor systems*. ACM Press, 2003.

[20] J. Polastre, J. Hill, and D. Culler. Versatile low power media access for wireless sensor networks. In *Proceedings of the 2nd ACM Conference on Embedded Network Sensor Systems*, 2004.

[21] J. Polastre, J. Hui, P. Levis, J. Zhao, D. Culler, S. Shenker, and I. Stoica. A unifying link abstraction for wireless sensor networks. In *Proceedings of the Third ACM Conference on Embedded Networked Sensor Systems (SenSys 2005)*.

[22] S. Ratnasamy, B. Karp, L. Yin, F. Yu, D. Estrin, R. Govindan, and S. Shenker. GHT: a geographic hash table for data-centric storage. In *Proceedings of the First ACM International Workshop on Wireless Sensor Networks and Applications*, 2002.

[23] C. Sharp, S. Schaffert, A. Woo, N. Sastry, C. Karlof, S. Sastry, and D. Culler. Design and implementation of a sensor network system for vehicle tracking and autonomous interception. In *Proceedings of the Second European Workshop on Wireless Sensor Networks (EWSN)*, 2005.

[24] S. Singh and C. S. Raghavendra. Pamas: power aware multi-access protocol with signalling for ad hoc networks. *SIGCOMM Comput. Commun. Rev.*, 28(3):5–26, 1998.

[25] The Institute of Electrical and Electronics Engineers, Inc. Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs), Oct. 2003.

[26] A. Woo, T. Tong, and D. Culler. Taming the underlying challenges of reliable multihop routing in sensor networks. In *Proceedings of the First International Conference on Embedded Networked Sensor Systems (SenSys)*, 2003.

[27] N. Xu, S. Rangwala, K. K. Chintalapudi, D. Ganesan, A. Broad, R. Govindan, and D. Estrin. A wireless sensor network for structural monitoring. In *Proceedings of the 2nd international conference on Embedded networked sensor systems (SenSys)*, 2004.

[28] W. Ye, J. Heidemann, and D. Estrin. An energy-efficient mac protocol for wireless sensor networks. In *Proceedings of the 21st International Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2002)*, New York, NY, June 2002.