6. Conclusions

Ad hoc networks are very unique in many respects and in general operate under highly challenging networking conditions. Various performance bottlenecks and network dynamics render communication difficult. Traditional network architectures such as the layered protocol stack do not sufficiently support and enable protocols to adapt to the dynamic networking conditions and do not offer mechanisms to relieve bottlenecks. Cross-layer mechanisms on the other hand allow protocols to obtain relevant information to adapt to changes efficiently and they can optimize their behavior accordingly.

Cross-layer adaptations so far were restricted to locally available information provided by protocols of the networking stack. Using this information protocol behavior can be adapted and optimized to achieve a desired objective such as energy conservation, throughput maximization, QoS support and many others. Relying solely on local information can lack accuracy and global, i.e. network-wide optimizations might not be achievable to a satisfactory level. The presented cross-layer architecture CrossTalk is able to provide locally available information and in addition it can create a network-wide view of the metrics available locally. CrossTalk also provides mechanisms to generate optimization metrics from available protocol data. In other words, CrossTalk can assist in generating meaningful data from very basic information which potentially as such is not very useful.

CrossTalk has many advantageous properties. One is that cross-layer enhanced protocols running inside of CrossTalk have to remain interoperable with their layered counterparts. This interoperability guarantees that CrossTalk can be introduced at any point in time even after a layered protocol stack has been deployed without having to change the protocol stack on each and every node. It also guarantees that if cross-layer information is not available such as GPS coordinates, the network remains fully functional as the layered mode can be resumed at any time. Many proposed solutions change the protocol itself, such as the protocol headers. This way interoperability is jeopardized.

By establishing fuzzy global knowledge at each node in the network, a node can evaluate its relative state. Based on this state, i.e. based on the comparison of its own local state with the network-wide state a node can perform lightweight actions locally achieving global objectives as shown in section 1.

CrossTalk per se allows protocols to store arbitrary data inside the Local View. That does not make much sense in most cases though as existing protocols will not be able to utilize this information. It is useful in the case that a global adaptation mechanism inside a protocol is based on network-wide information from that protocol itself. In general CrossTalk provides some basic metrics that can be expected to be available such as load, topological mobility and some others. CrossTalk provides some data abstractions that can be utilized to represent data such as routing state and a unified interface to access it. There are no advanced features such as semantics to describe and interpret the data offered which would make the architecture unnecessarily complicated and hinder a widespread adoption and the potential longevity of the architecture.
As shown in the results section, adaptations based on CrossTalk can be very simple and often do not require strong protocol redesign. Therefore, the process of creating cross-layer protocol extensions is very light-weight and straight forward.
CrossTalk also provides a mechanism to prevent and detect harmful cross-layer interactions and adaptations. Simple protocol signaling can force a protocol to solely rely on layered functionality. This way, if the cross-layer adaptation makes the network dysfunctional a protocol can detect this and resolve the problem. This mechanism can also be used to compare the performance between the layered and the cross-layer approach at runtime.
CrossTalk was exemplary applied to perform load balancing, mobility adaptations and it was shown that its Global View component can support and enable novel applications efficiently. The establishment of a network-wide view was analyzed according to its scalability properties and its quality with highly satisfactory results. The exemplary applications showed that CrossTalk is able to support adaptations and optimizations efficiently by significantly increasing protocol performance when compared to the layered counterparts.
In chapter 3.4 related architectures were compared against each other. The remainder of this chapter is used to add CrossTalk to this comparison.

Application generality: CrossTalk does not assume a scenario for its application. It does not assume a wireless sensor network, a backbone infrastructure or a special operating system to be present as many presented architectures do.

Functional complexity: The functional complexity of CrossTalk is very low. Cross-layer mechanisms are very straight forward and have a very small footprint as shown in section 1.

Data management complexity: CrossTalk's Global View component and the metric generator introduce some functional complexity. But in general this complexity is very low as it is very straight forward data management and simple algorithms. In addition, the overall architecture itself is of a very low complexity when compared to other approaches.

Network optimization potential: The network optimization potential is very high using CrossTalk. The reason is that CrossTalk not only utilizes local information but also allows protocols to have a network-wide view. CrossTalk also has no restrictions on data read access.

Protocol complexity: Many of the architectures proposed were never actually utilized to optimize protocol behavior. Only judging from the general descriptions and the components involved many architectures force a fundamental protocol redesign. CrossTalk on the other hand found many applications already. Every optimization and adaptation realized using CrossTalk only needed minor changes to the original protocols. Therefore, CrossTalk quite obviously only requires a very small amount of effort to be usefully applied.

Assumptions and preconditions: Many proposed architectures make certain assumptions about the suitability of protocols or about the hard- and/or software requirements. CrossTalk does not make such assumptions. The architecture is held very general and in the results section protocols were chosen that other architectures believe are less suitable. Still by applying CrossTalk significant performance gains were achieved.

Table 6.1 Comparison of CrossTalk and related architectures

Criterion / Architecture	Application generality	Functional complexity	Data management complexity	Network optimization potential	Protocol complexity	Assumptions and preconditions
CATS	-	-	-	+	0	-
ÉCLAIR	++	--	-	+	--	0
GRACE	+	-	--	+	-	-
MobileMan	++	+	0	+	+	-
WIDENS	0	0	-	0	-	-
TinyCubus	--	-	-	0	0	-
CrossTalk	++	0	0	++	+	0