# Protocol Factory: Reuse for Network Experimentation

Georg Kunz, Olaf Landsiedel, Stefan Götz, Klaus Wehrle
Distributed Systems Group
RWTH Aachen University, Germany
*firstname*.*lastname*@cs.rwth-aachen.de

## 1. INTRODUCTION AND OVERVIEW

Today's protocol research and development process is often enough extremely labor-intensive: First, existing protocol implementations do not allow to flexibly modify and extend their functionality. This forces researchers to reimplement common protocol functionality for each new protocol. Second, protocol implementations are highly platform depend. Again, reimplementions become necessary when migrating implementations between different evaluation platforms for analysis.

This poster proposal introduces Protocol Factory (ProFab), a protocol composition framework for rapid prototyping and network experimentation. ProFab relies on two contributions: First, a library of generic functional building blocks for composing full-fledged protocol stacks. Second, a virtual platform that embeds protocols on a wide range of evaluation platforms such as simulators, testbeds and OS kernels. The remainder of this poster proposal illustrates these two key design aspects of ProFab in more detail.

## 2. GENERIC PROTOCOL LIBRARY

Within and across layer boundaries, protocols provide similar services. Our analysis indicates that a specific protocol is largely determined by the composition of those services and their configuration. Thus, by decomposing protocols into configurable building blocks and enabling their composition and re-use, ProFab effectively reduces the protocol development effort.

ProFab allows composing functional building blocks dynamically at run-time and statically at compile-time. The first mechanism enables elegant handling of flows and sessions by instantiating services. In contrast, static composition aims for performance as it eliminates the run-time overhead of modularization and instantiation by means of a pre-compiler. In both cases, the pre-compiler customizes the functional building blocks according to a given configuration. This configuration process adapts the generic building blocks to efficiently incorporate the semantics of a particular protocol.

We implemented an entire protocol stack including ARP, IPv4/6, and IPX as well as TCP, DCCP, and SCTP. We found that both network layer and transport layer protocols indeed share approximately 70% of their modules. Moreover, we believe that similar sharing ratios are achievable among application layer protocols – in particular those that exhibit transport layer functionality such as streaming protocols.

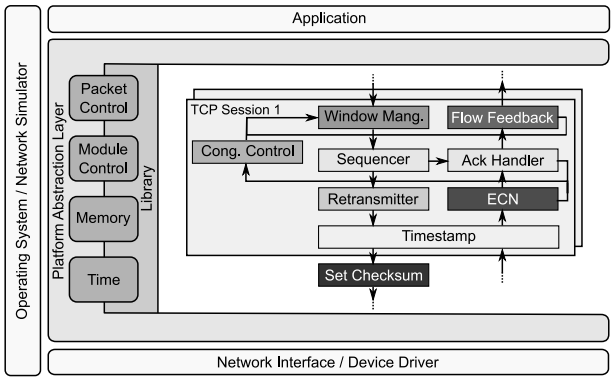


**Figure 1: The virtual platform encapsulates modular protocol stack compositions**

## 3. VIRTUAL PLATFORM

Protocols require access to system specific resources such as memory, timer and network data. Our virtual platform bases on the observation that system design has evolved best practices among execution models and system APIs. Hence, these similarities provide a narrow waist for platform abstraction, allowing the virtual platform to remain lightweight, yet complete (see Figure 1).

A key enabler for a lightweight virtual platform is a unified execution model that is natively supported on all platforms. ProFab employs asynchronous event-based execution, which directly maps to interrupt handling in operating systems and testbeds as well as event scheduling in simulation frameworks. Similarly, C constitutes the joint programming language. It is available on nearly all platforms including OS kernels and embedded systems and allows efficient access to system resources.

The virtual platform covers a wide range of systems (Linux & Windows XP/CE user- and kernel-space, TinyOS, ns-2/3, OMNeT++) with just approximately 1000 lines of code each. Moreover, performance evaluations comparing ProFab's modular architecture to native Linux and Windows systems indicate a minor impact on performance.

## 4. CONCLUSION

Already during its development, ProFab's two-fold approach and its carefully design architecture proofed to be an valuable addition to the protocol development process. The proposed poster puts a special focus on the key design challenges, the resulting architecture of ProFab, and our evaluation results.