2 Introduction

This document is intended to assist software developers who are either implementing MCAPI or writing applications that use MCAPI.

The MCAPI specification is both an API and communications semantic specification. It does not define which link management, device model or wire protocol is used underneath it. As such, by defining a standard API, it is intended to provide source code compatibility for application code to be ported from one operating environment to another.

MCAPI defines three fundamental communications types. These are:

1. Messages – connection-less datagrams.
2. Packet channels – connection-oriented, uni-directional, FIFO packet streams.
3. Scalar channels – connection-oriented single word uni-directional, FIFO scalar streams.

Each of these communications types have their own API calls and are desirable for certain types of systems. Messages are the most flexible form of communication in MCAPI, and are useful when senders and receivers and per message priorities are dynamically changing. These are commonly used for synchronization, initialization and load balancing.

Packet and scalar channels provide light-weight socket-like stream communication mechanisms for senders and receivers with static communication graphs. In a multicore, MCAPI's channel APIs provide an extremely low-overhead ASIC-like uni-directional FIFO communications capability. Channels are commonly set up once during initialization, during which the MCAPI runtime system attempts to perform as much of the work involved in communications (such as name lookup, route determination, and buffer allocation) between a specific pair of endpoints as possible. Subsequent channel sends and receives thus incur just the minimal overhead of physically transferring the data. Packet channels support streaming communication of multiword data buffers, while scalar channels are optimized for sequences of scalar values. Channel API calls are simple and statically typed, thereby minimizing dynamic software overhead, which allows applications to access the underlying multicore hardware with extremely low latency and energy cost.

MCAPI's objective is to provide a limited number of calls with sufficient communication functionality while keeping it simple enough to allow efficient implementations. Additional functionality can be layered on top of the API set. The calls are exemplifying functionality and are not mapped to any particular existing implementation.