INTRODUCTION

Many tracing and monitoring tools share similar structure: various sources such as processes or management-enabled hardware devices generate data and input it to an aggregation tree, which collects the information and transmit it to one or more endpoints. Along the way, data may be temporarily or permanently cached.

Our goal is to explore the trade-offs in this design space, and offer users a flexible system for implementing this type of application which abstracts away some of the difficulties. We incorporate the key ideas of mobile code and declarative specification in our system design.

BENEFITS and USE CASES

Users and Use cases:
• RoR app: unified monitoring of web queries, load balancer, and machine load for on- or offline processing.
• X-Trace: replace heavyweight JMS with declarative spec and small daemon.
  • Module Container runs on each machine to collect local reports.
  • Reports buffered, transmitted to collection point.
• Log post-processing: exploit lack of distinction between running modules and stored data.

Benefits:
• Shared code in framework speeds development.
• Common abstractions mean modules are reusable, and existing modules can be composed to build new tools.
• Declaratively driven deployment.

RELATED WORK and CONTEXT

• Message Oriented Middleware/PubSub architectures: heavyweight back ends and large activation energy to get something running.
• Ganglia: collect and aggregate telemetry on order thousand nodes.
• P2: provides streaming database queries, declarative overlays.
• Other management platforms: Systems like Tivoli and OpenView are large, complex, difficult to use, and costly. Not suited for research or startups.

CPU Overhead of Remote Messaging

ROADMAP

General:
• Visualization tools
• Output to external processes
• Replay of experiments to support agile development of ML-based monitoring and control tools in the lab.

To support X-Trace:
• Reliable buffering and replay
• Transparent stream compression (and encryption?)
• High reliability and robustness
• Controlled resource consumption
• Database/relational store interface