

# Leveraging the Eclipse™ TPTP\* Agent Infrastructure

Andy Kaylor  
Intel Corporation  
[andrew.kaylor@intel.com](mailto:andrew.kaylor@intel.com)

\*Test and Performance Tools Platform (formerly known as Hyades)

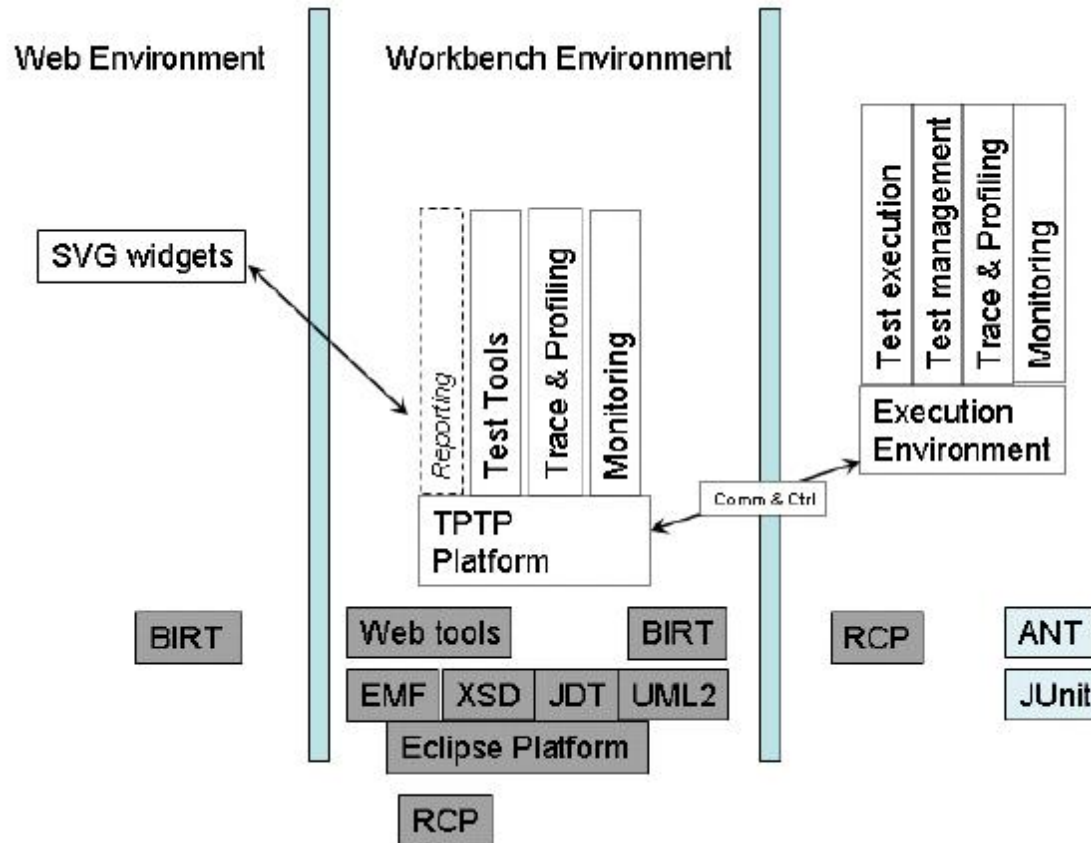
## Goals for this Session

- Provide an introduction to the TPTP Agent Infrastructure
- Explain the benefits of developing systems based on this infrastructure
- Explore the technical details of TPTP-based agent development

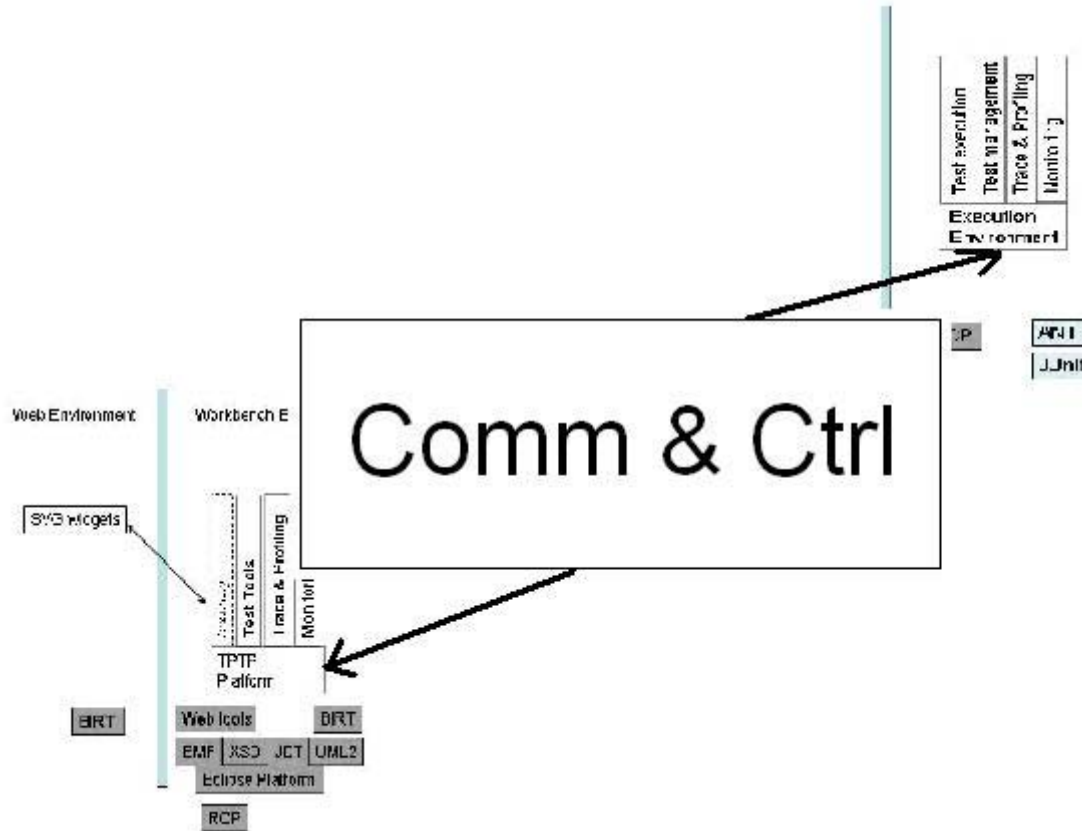
## What is TPTP?

- Test and Performance Tools Platform Project
- TPTP is an integrated testing, tracing, profiling and monitoring platform.
- It encompasses everything from data collectors to a data model and viewers integrated with Eclipse.
- Although TPTP includes a set of exemplary tools, it is intended as a platform upon which other software test and performance tools are built.

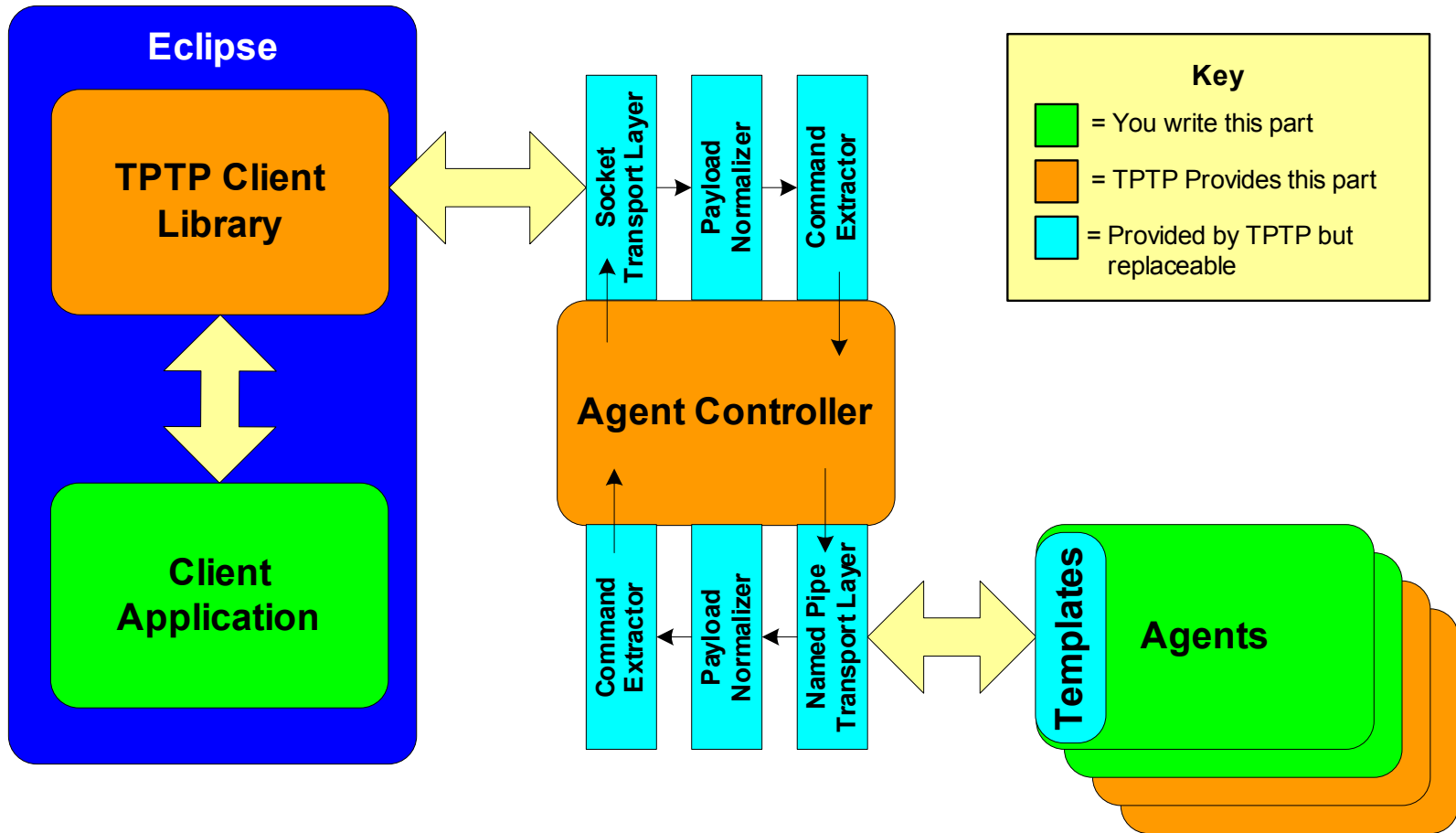
# The Official TPTP Architecture



# The Way It Looks To Me



# A View from 10,000 Feet

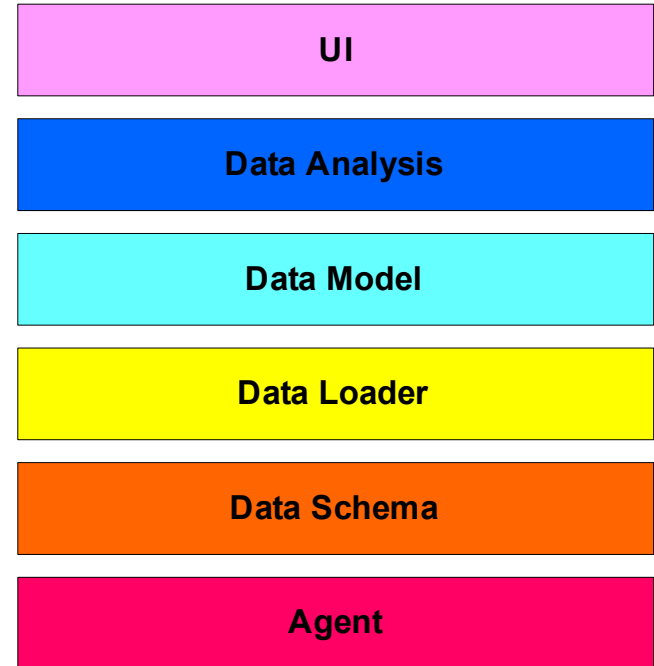


## Why Use TPTP?

- TPTP provides a ready-to-use, but extensible, solution for data collection and testing development
- TPTP is an open source, collaborative platform
- Eclipse integration provides a solid foundation for UI development
- The common framework provides the opportunity for your product to be used seamlessly alongside other tools
- The standards-based common agent interface allows you to make your data collector available to other tools

## Points of Entry

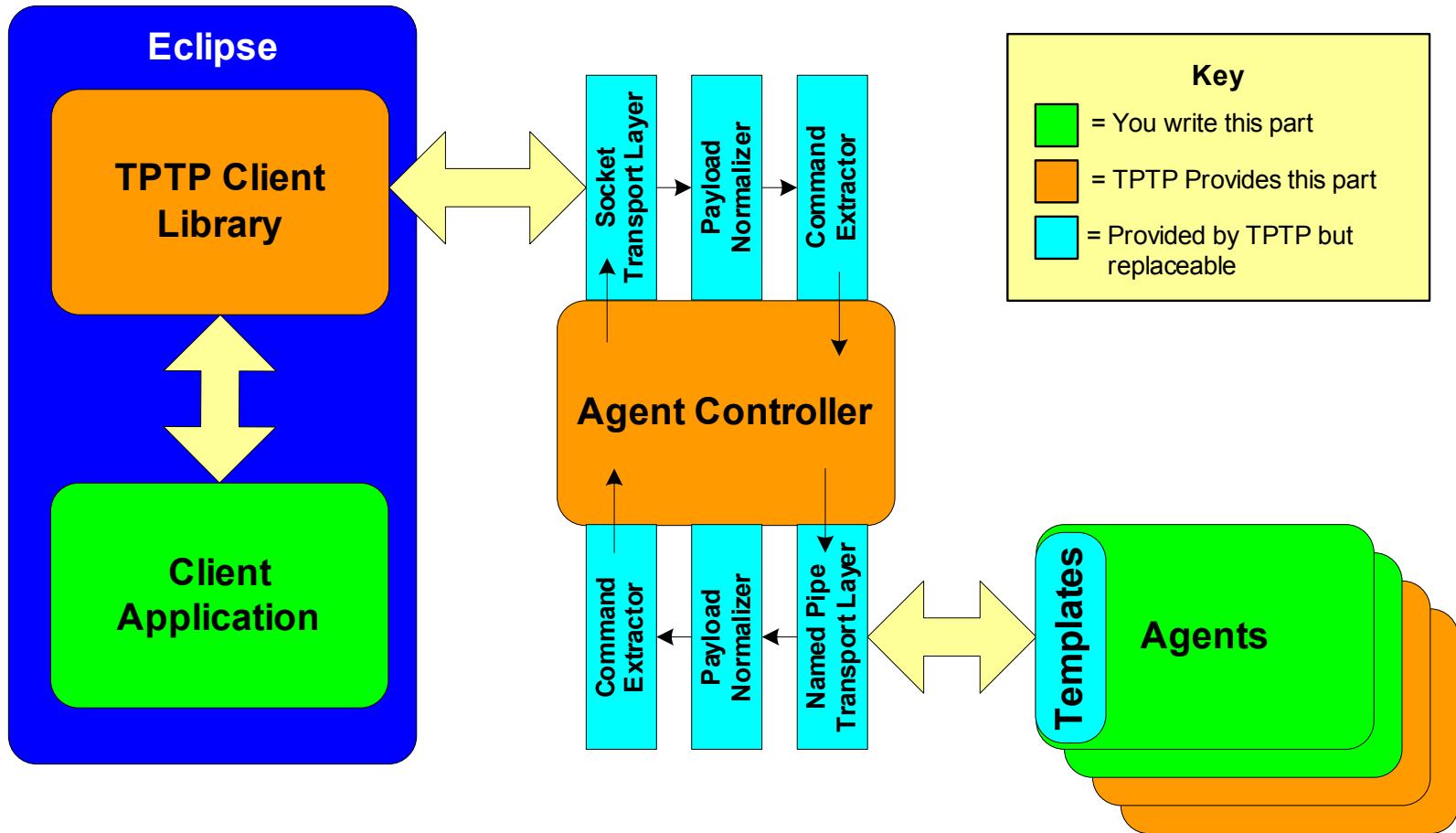
- TPTP is designed as a stack of possible integration points
- Each layer provides a possible point of entry for integration
- Using standard interfaces, you can provide one or more layers while leveraging existing modules for the other layers



## What is an Agent?

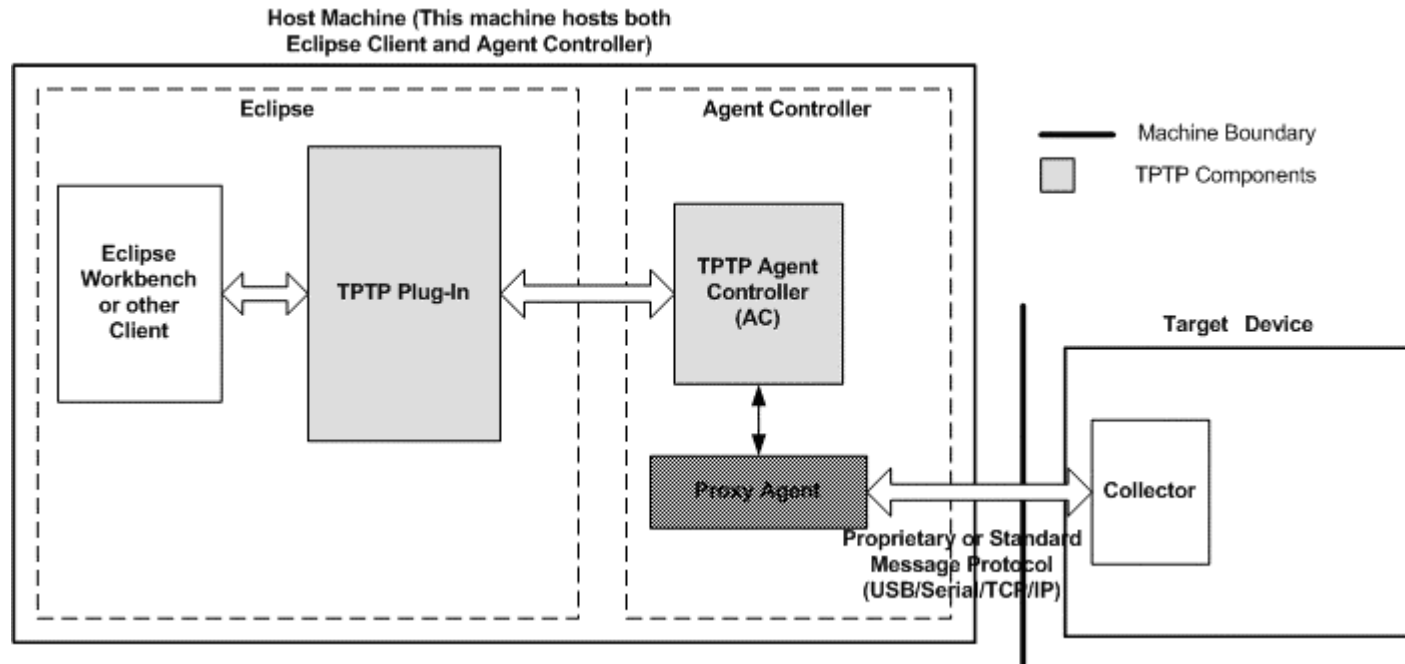
- Within the context of TPTP, an agent is defined as a logical object that exposes services through the TPTP Agent Controller
- Data collectors are an import type of agent, but the concept is generalized in TPTP
- Other agents can provide services such as file transfer or system information
- Access to categories of agents, such as data collectors, is generalized using common interfaces

# The 10,000-Foot View Again



# Supporting Small Scale Devices

- In some cases, it may be preferable to access your device through a proxy



## Interfaces and Commands

- Interfaces and commands are the basic building blocks of the protocol TPTP uses for communication between components
- Each command has a piece which the Agent Controller recognizes and a piece which is command-specific
- Command details are defined by the interface to which the command belongs
- A standard set of interfaces is defined, but you are expected to define your own interfaces to extend the protocol

## Finding an Agent

- Clients locate agents by sending requests to the Agent Controller
- The Agent Controller will launch agents as needed
- Agents may also be launched using other mechanisms as needed
- Agents can be located by name or by supported interfaces
- Agents may also provide additional metadata which can be used to locate the agent of interest

## Agent Development

- Agents are implemented as standalone processes which communicate through the Agent Controller.
- Agents are driven by messages exchanged either with a client application or another agent.
- TPTP provides libraries to handle the boilerplate tasks of agent development, such as command parsing and communications
- Agent development is supported for C, C++ and Java

## Custom Interfaces

- The development toolkit provides direct support for implementing standard interfaces and the associated commands
- Client-side classes are also provided for sending standard commands
- In order to extend the protocol with custom interfaces, you will need to write some additional code to read and write these commands

## Agent Variables

- One of the standard interfaces specified by TPTP is an interface designed to expose configurable variables within an agent
- Agents which have variables that can be set at run-time should expose them through this interface
- This common interface will enable TPTP to provide a generic client-side interface for configuring the agent

## Feed the Data Models

- The TPTP user experience is centered around EMF data models
- Sharing a common data model maximizes the possibility for interoperability between tools
- Data written into the standard data models can be consumed by existing viewers
- If your agent outputs data in the expected XML format, TPTP-provided data loaders will put the data into the data model
- If you have a custom format, you can write your own data loader to put data into the data model

## Key Points

- TPTP provides a platform for local or remote data collection and testing development
- TPTP supplies boilerplate code, allowing you to focus on your own functionality
- TPTP is an extensible, open source platform

Questions?