



The Art of: Pair Programming

Presented By: Todd Merritt

Todd Merritt



Email: TLMerritt@Gmail.com.com

Twitter: @GeekInterface

LinkedIn:

<https://www.linkedin.com/in/tlmerritt/>

- Over 16 years of Development Experience
- Over 6 years Pair Programming Experience
- Worked with Small Startups to fortune 500 companies
- Interests:
 - App Design/Development
 - Database Development
 - DevOps – Sometimes


DevSpace would like to thank our sponsors



Disclaimers:

(Pairing \neq )

(Pairing $\equiv \equiv$)

A hammer with a black head and a wooden handle with a red band.



WHEN YOU DON'T PAIR

It makes pandas sad

Assumptions:



Coding Is Social

A photograph of two people in an office setting. In the foreground, the back of a person's head and shoulders are visible; they are wearing a dark and light checkered shirt. To their left, another person is partially visible, looking towards a computer monitor. The monitor displays a code editor with syntax-highlighted text. The office has a modern feel with a wooden slatted ceiling and glass partitions in the background.

Pairing is a Discussion



The speed of programming is
limited by thought not typing

(typeof(Programming) == "Work")

(Pair Programming == Pair Working)



The speed of **WORK** is
limited by the speed of thought



Pairing is not just for Programming

Pair Programming is a development technique where two programmers work together at the same work station at the same time.

Pair Programming

((Programmer x 2) + WorkStation + Discussion)

Benefits of Pairing

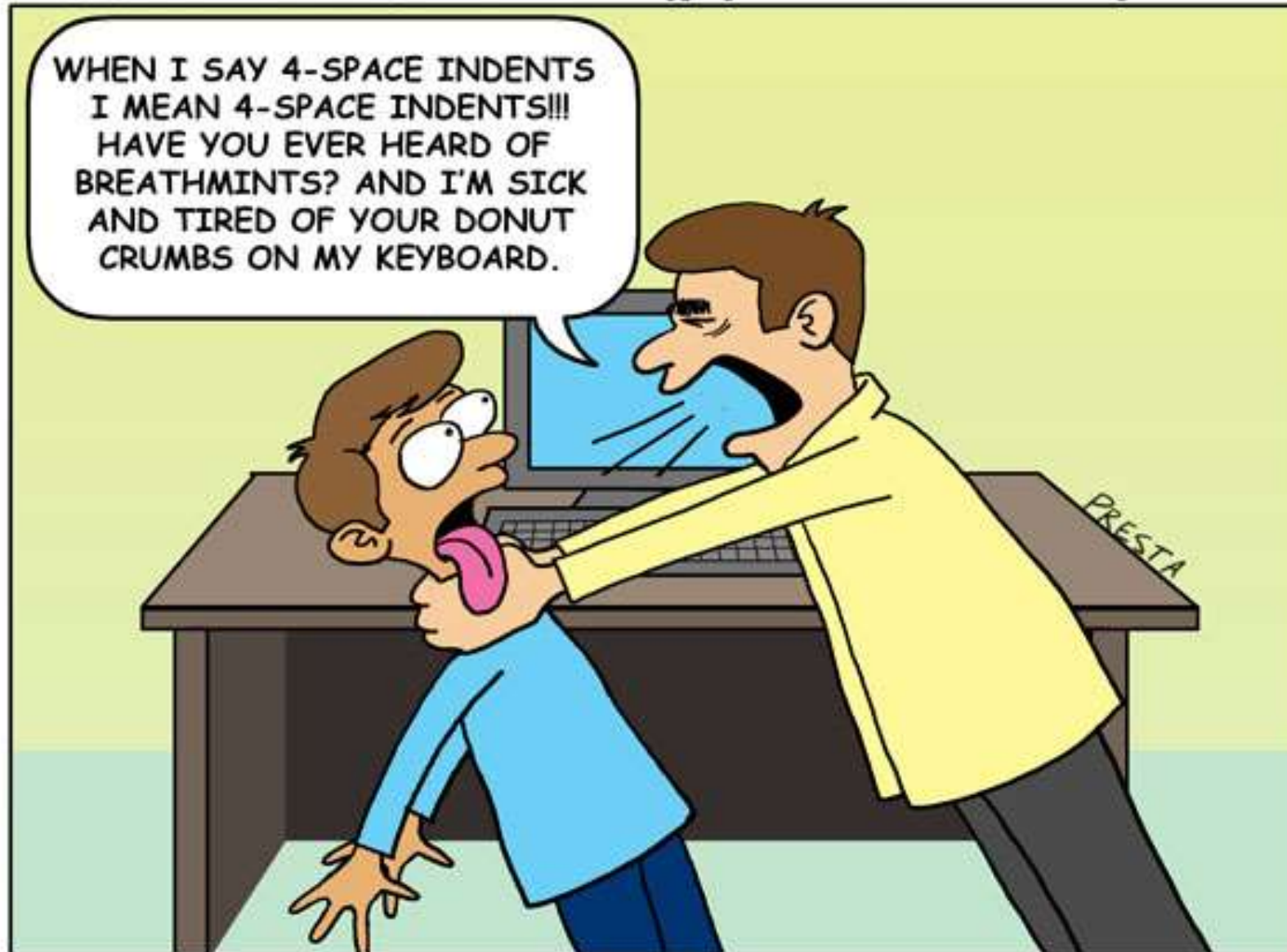
- Collective Code Ownership
- Team Building
- Reduction in Calendar Time
- Knowledge Transfer
- Focus on Quality over Quantity
- Prevents Silos
- Constant Code Reviews
- Quick Turnaround on Feedback
- Prevents Technical Debt
- Reduces Poor Programming Practices

Ideal Pairing Partner

- Good Communicator
- Willingness to Fail
- Asks Good Questions
- Accepts Productive Criticism
- Controls Ego
- Similar Schedules
- Trusts Partner

Potential Issues with Pairing

- Personal Space
- Cultural Misunderstanding
- Scheduling
- EGO/Attitudes
- Change Resistant
- Control Freak
- Lack of Communication



The dark side of pair programming.

Things to Discuss when Pairing

- Logic
- Design
- Refactoring
- Naming
- Order of Operation
- Requirements
- Anything related to System

When to Pair



Complex Systems



Mission Critical Systems



Constant Change/ Volatile Systems



Shared Code\Libraries



High Risk Deadlines



Knowledge Transfer / Mentoring



Onboarding Team Members

Common Reasons to Pair

- Fewer Defects
- Simpler Designs
- Faster Problem Solving
- Rapid Feed Back
- Knowledge Transfer
- Better Communication
- Enjoyable

When Not To Pair

- Simple Tasks
- Non-Production Code Spikes
- Partner Is Sick (**Only Pair Virtually**)



Pairing is
Instinctive?



Code Reviews



System Architecture Design

Analyzing Systems



Mentoring



New Technologies and Methodologies

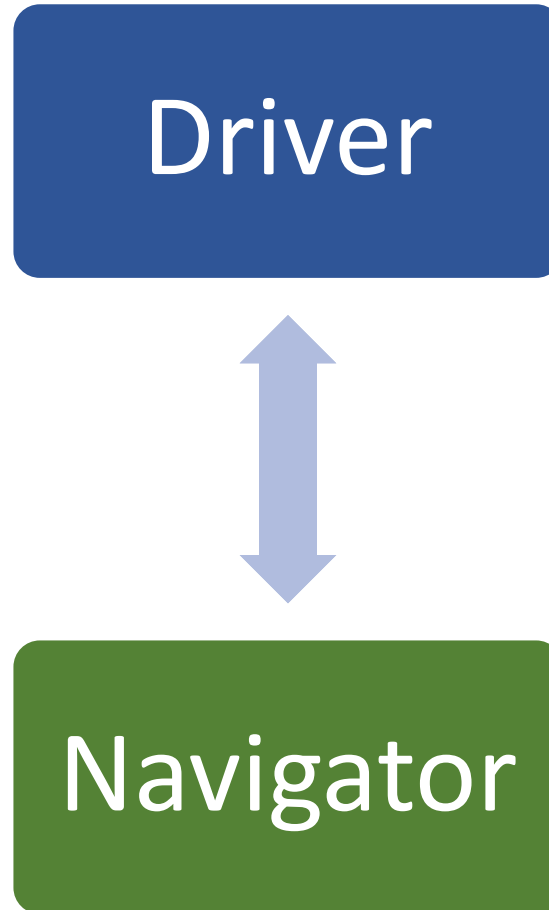


Production Support

A photograph of two women in an office environment. The woman in the foreground, with blonde hair, is looking down at a laptop screen with a slight smile. Another woman with blonde hair is leaning over her shoulder, also looking at the screen. In the background, a man with glasses is working on a laptop. A blue horizontal bar is overlaid across the middle of the image, containing the text 'Pairing Roles' in white.

Pairing Roles

Pairing Roles



Pairing Roles

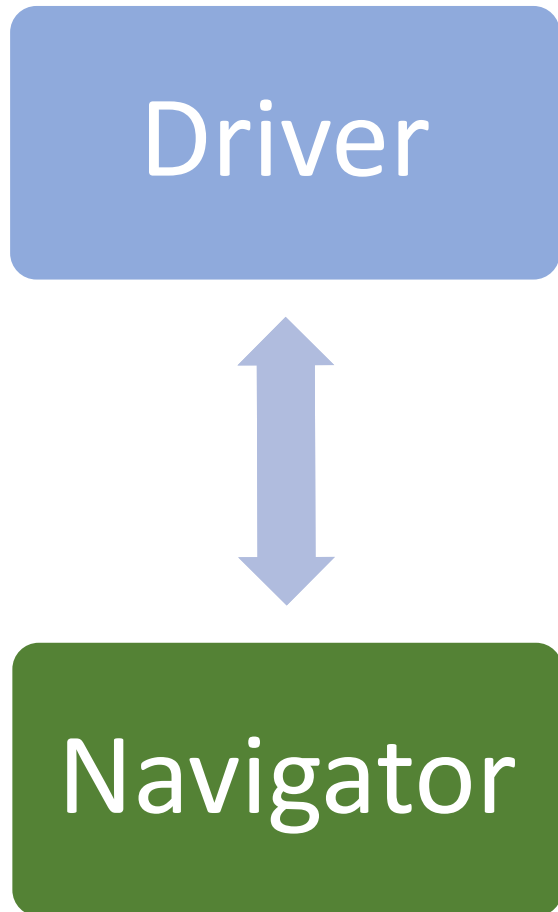
Driver



Navigator

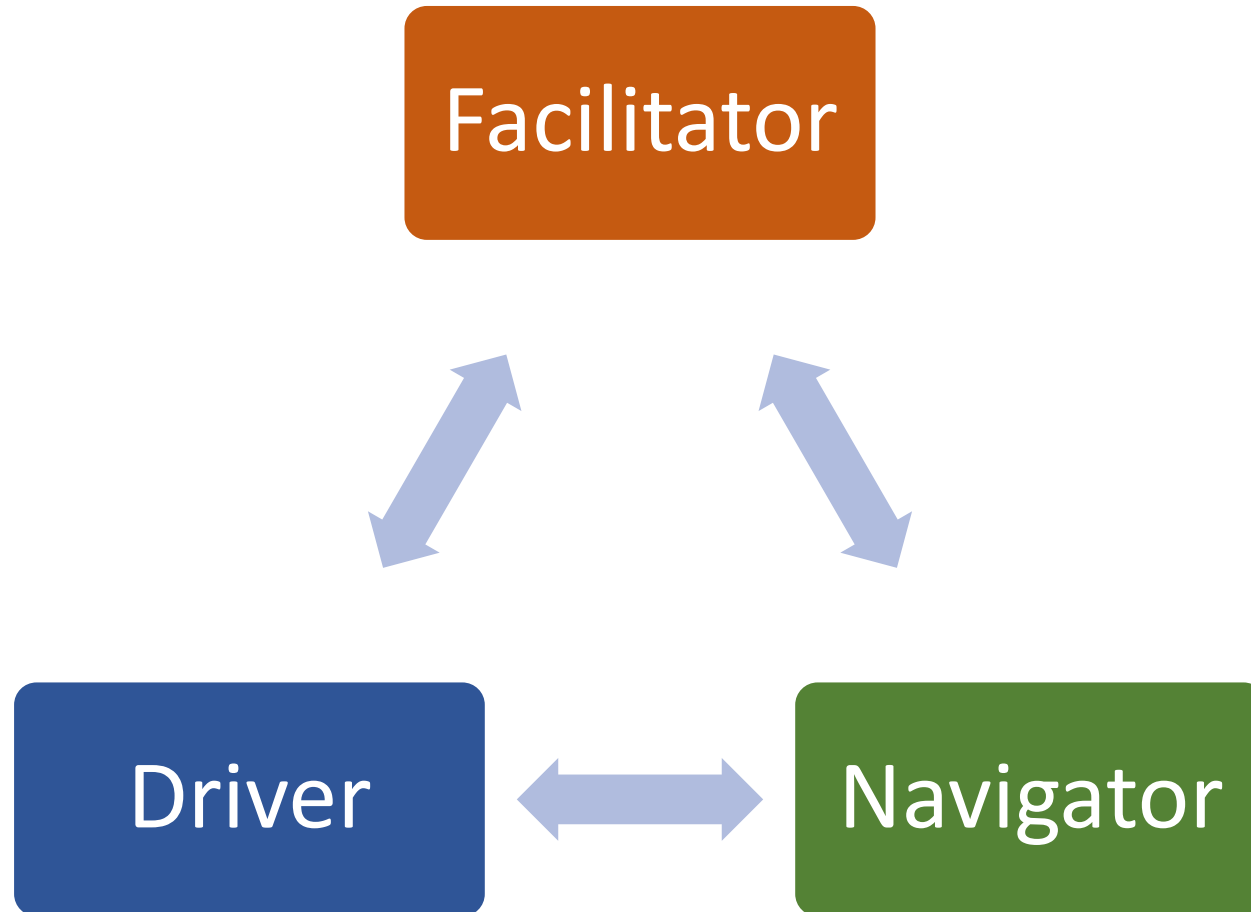
- Controls Keyboard and Mouse
- Discuss Ideas and Concepts being coded
- Constantly communicating with Partner

Pairing Roles

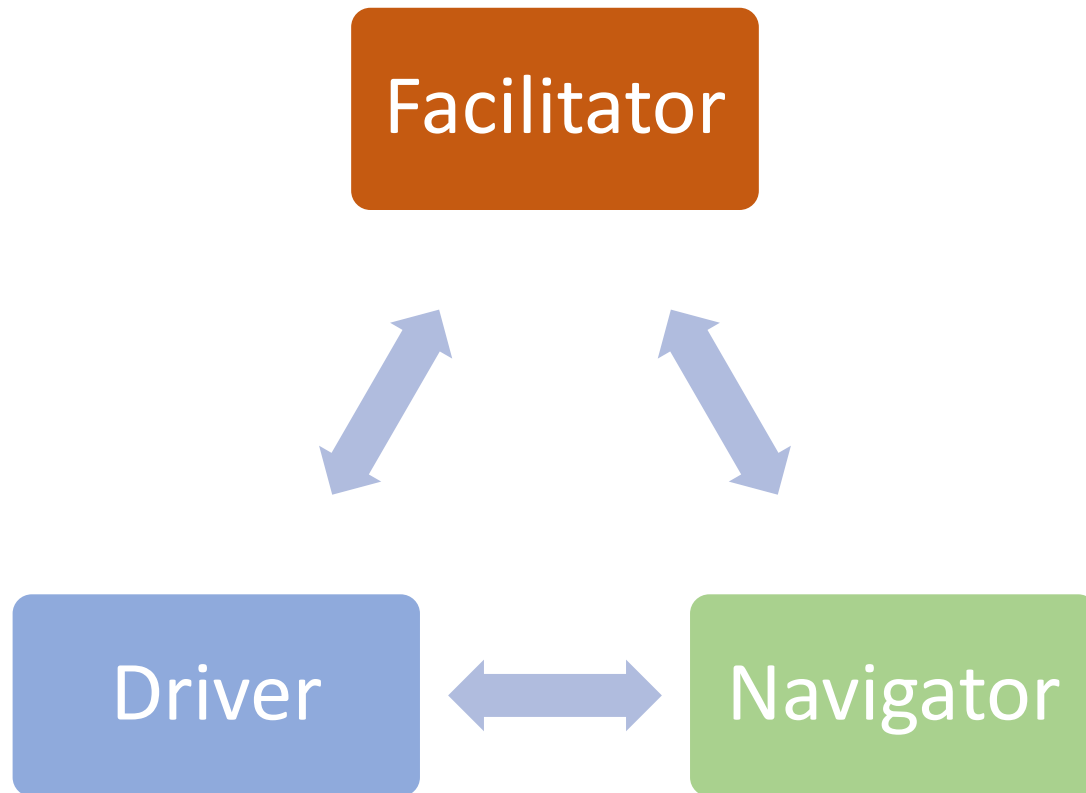


- Reviews Code
- Thinks Big Picture
- Discuss Ideas and Concepts being coded
- Constantly communicating with Partner

Asynchronous Pairing Roles



Asynchronous Pairing Roles



- Performs sub tasks related to the primary development task.
- Constantly communicating with Driver
- Constantly communicating with Partner

Switching Roles



Time Box Switching

A close-up photograph of a person's arm and hand holding a red ping pong paddle. The person is wearing a white long-sleeved shirt with the sleeves rolled up. They are standing at a white table, likely a ping pong table. In the background, another person in a dark blue shirt is visible, also holding a paddle, but they are out of focus. The scene is indoors with warm lighting.

Ping Pong Switching



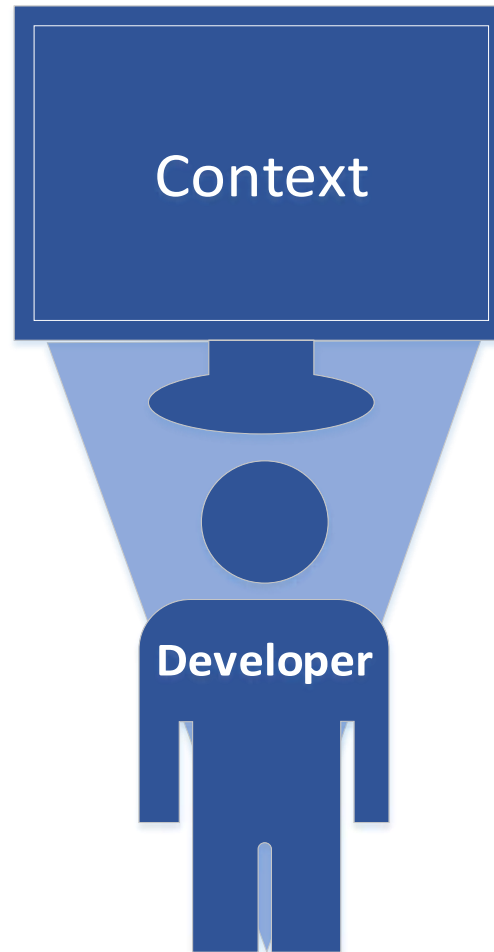
Ad Hoc Switching

Pairing Styles



Unpaired

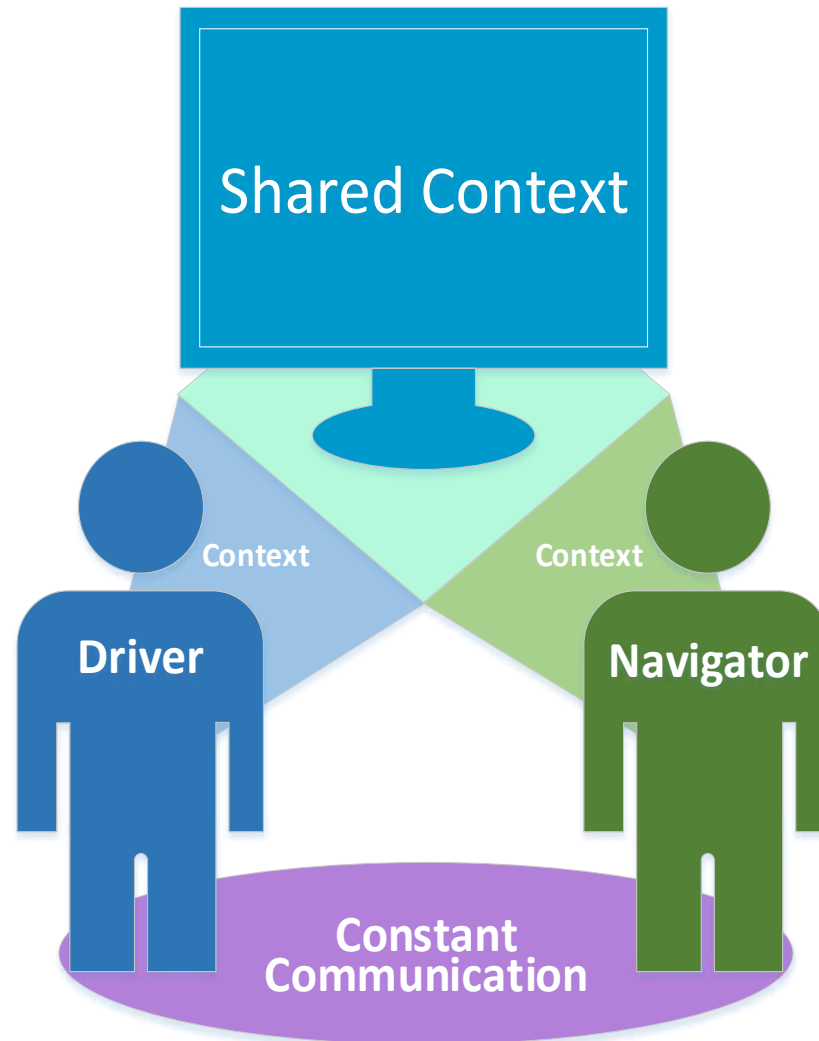
Unpaired





Standard Pairing

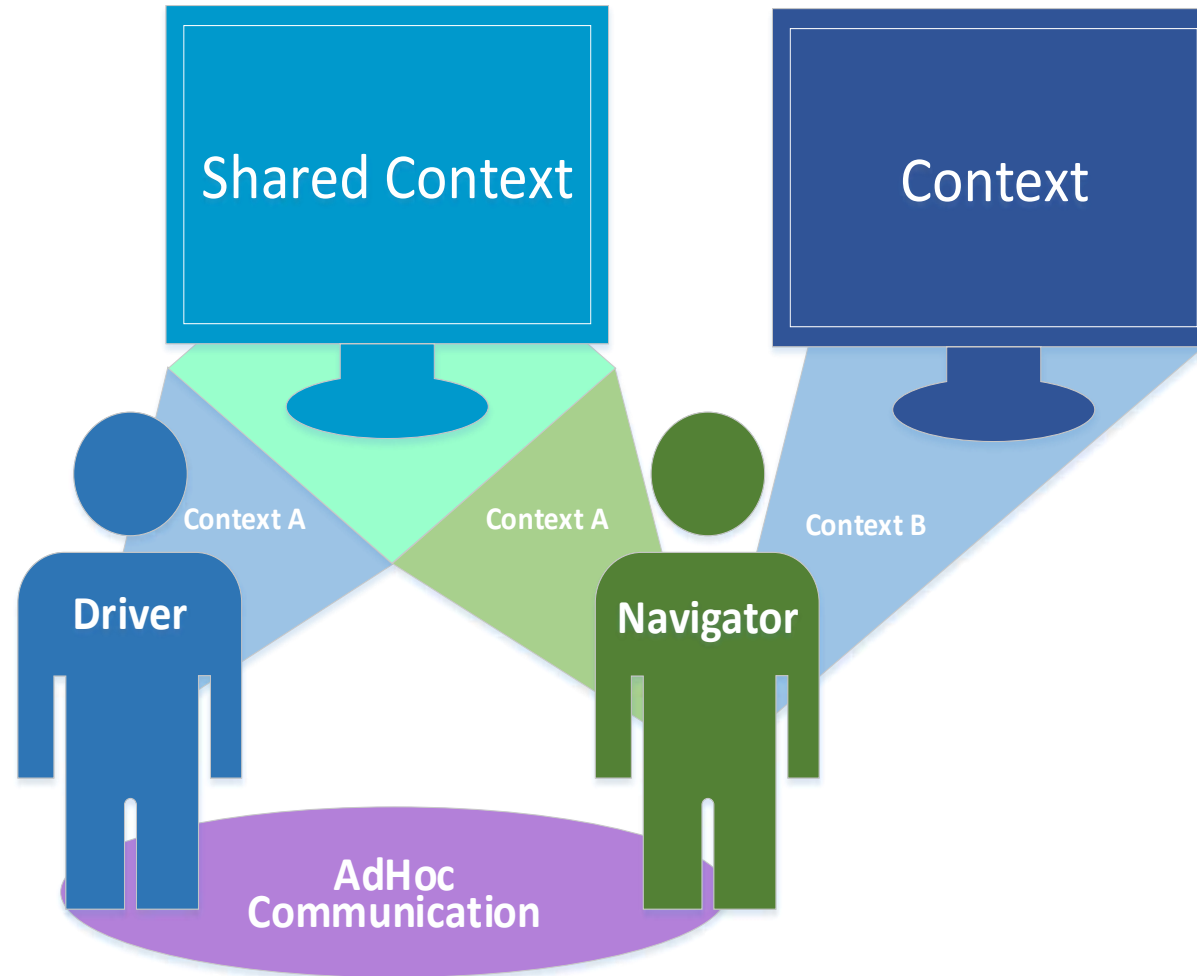
Standard Pairing





On Demand Pairing

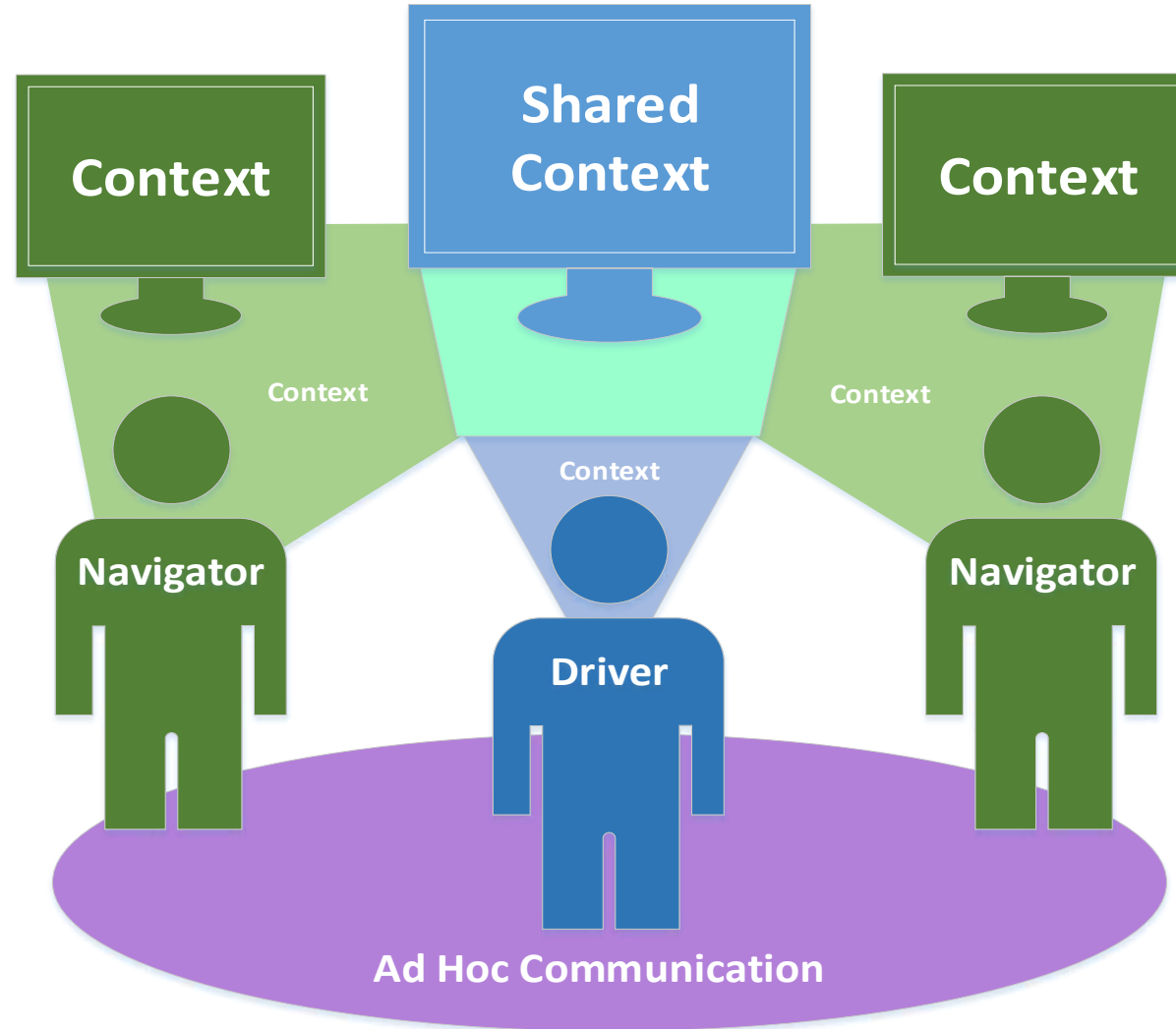
On Demand Pairing





Mob Paring

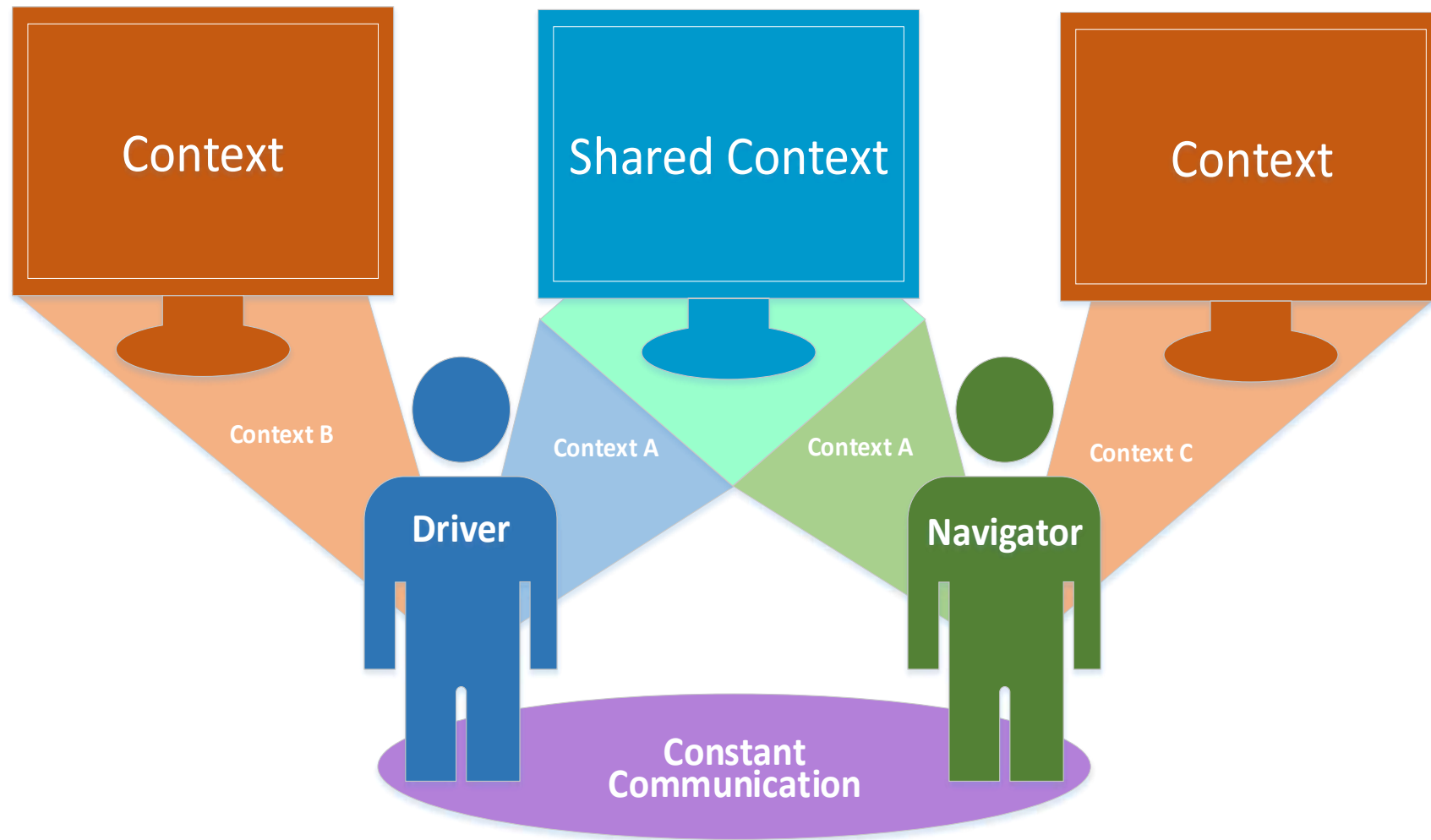
Mob Pairing



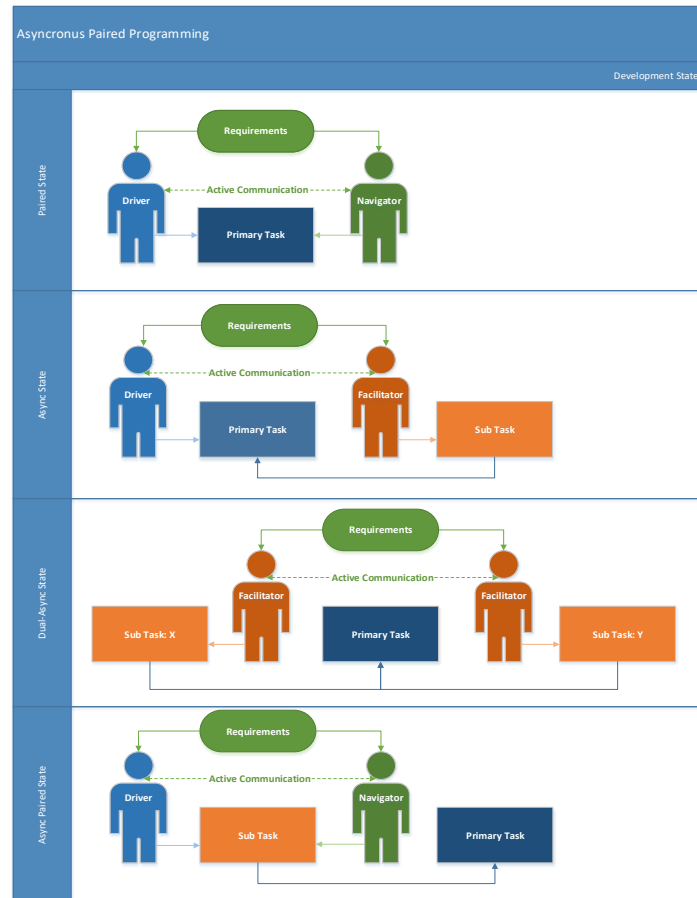


Asynchronous Paring

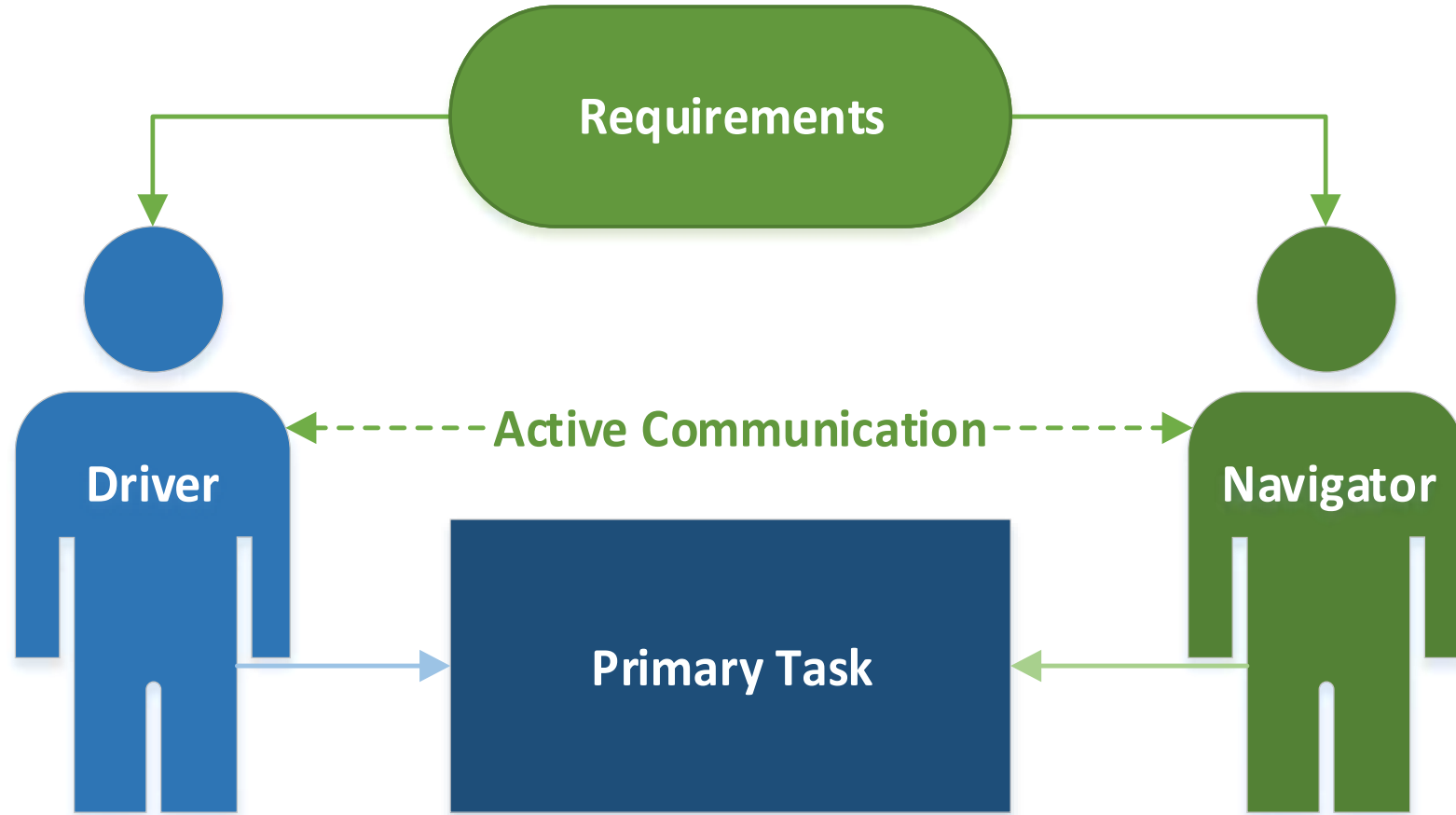
Asynchronous Pairing



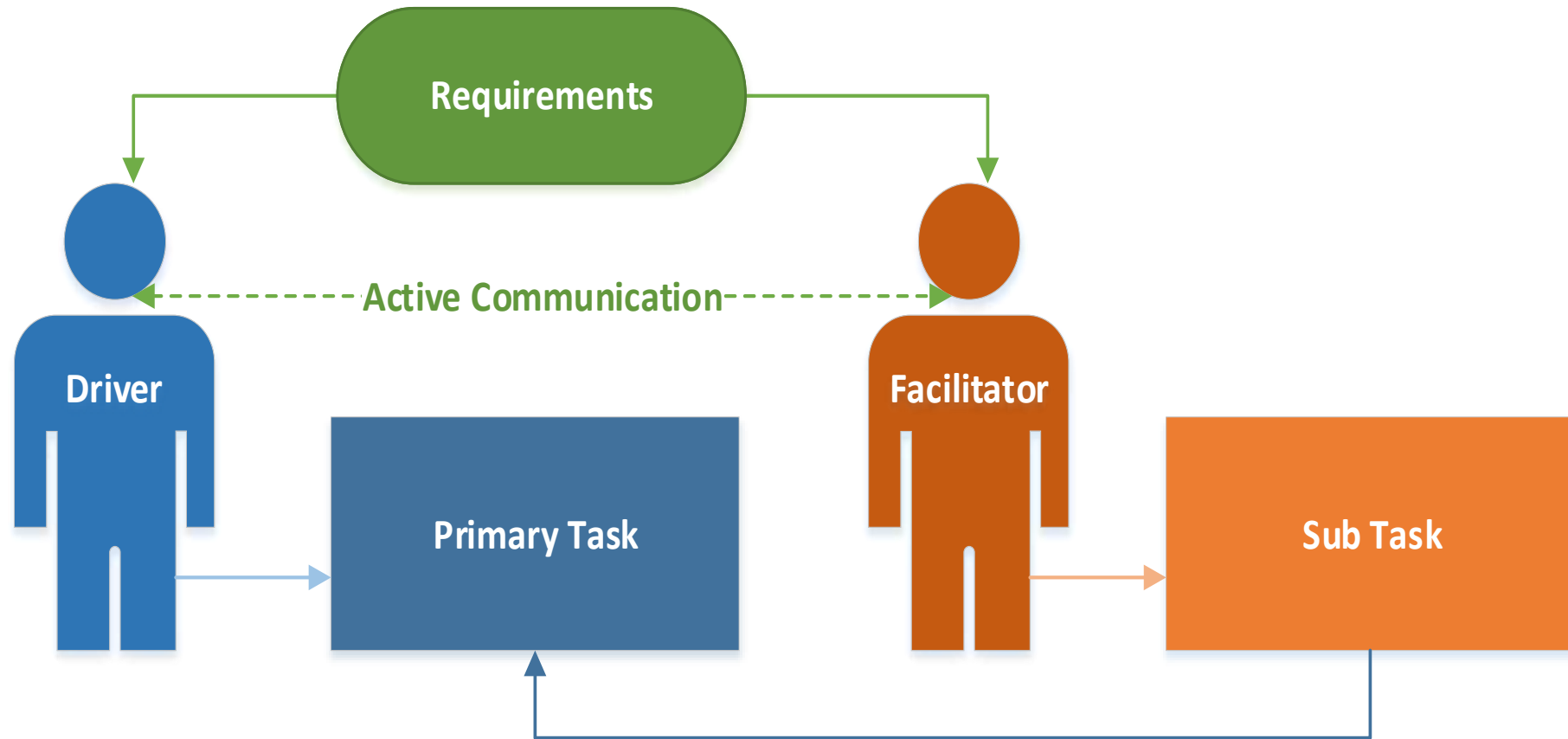
Asynchronous Pairing States



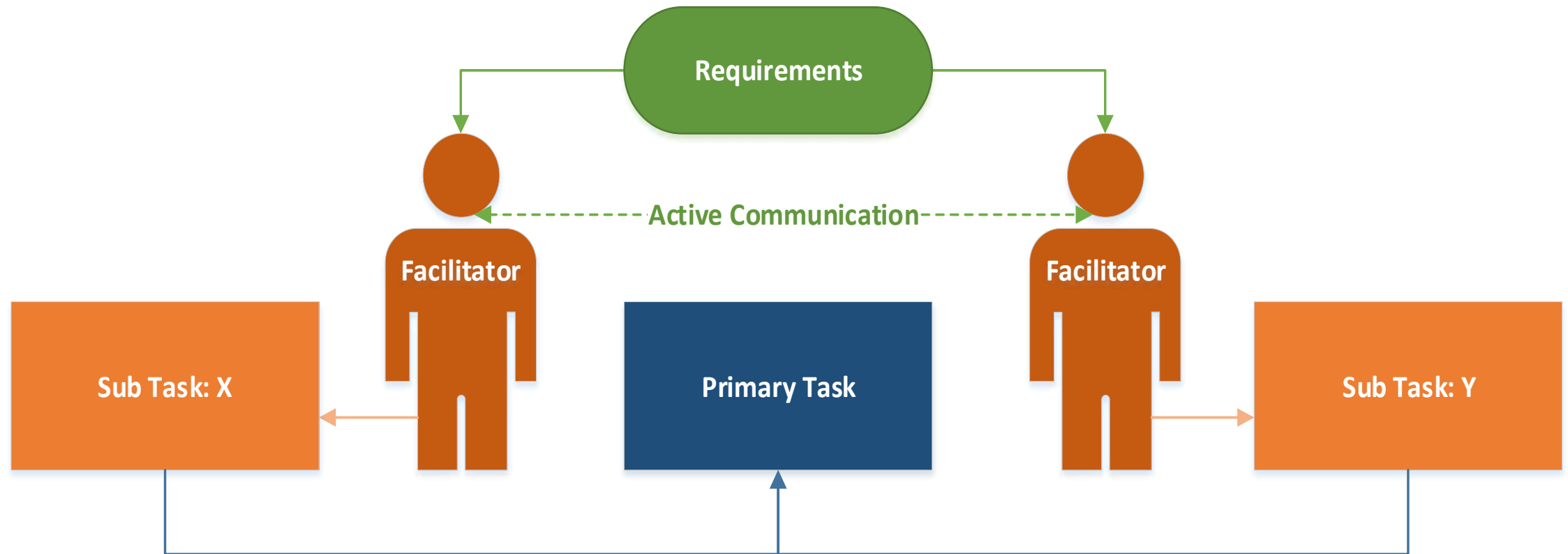
Paired State



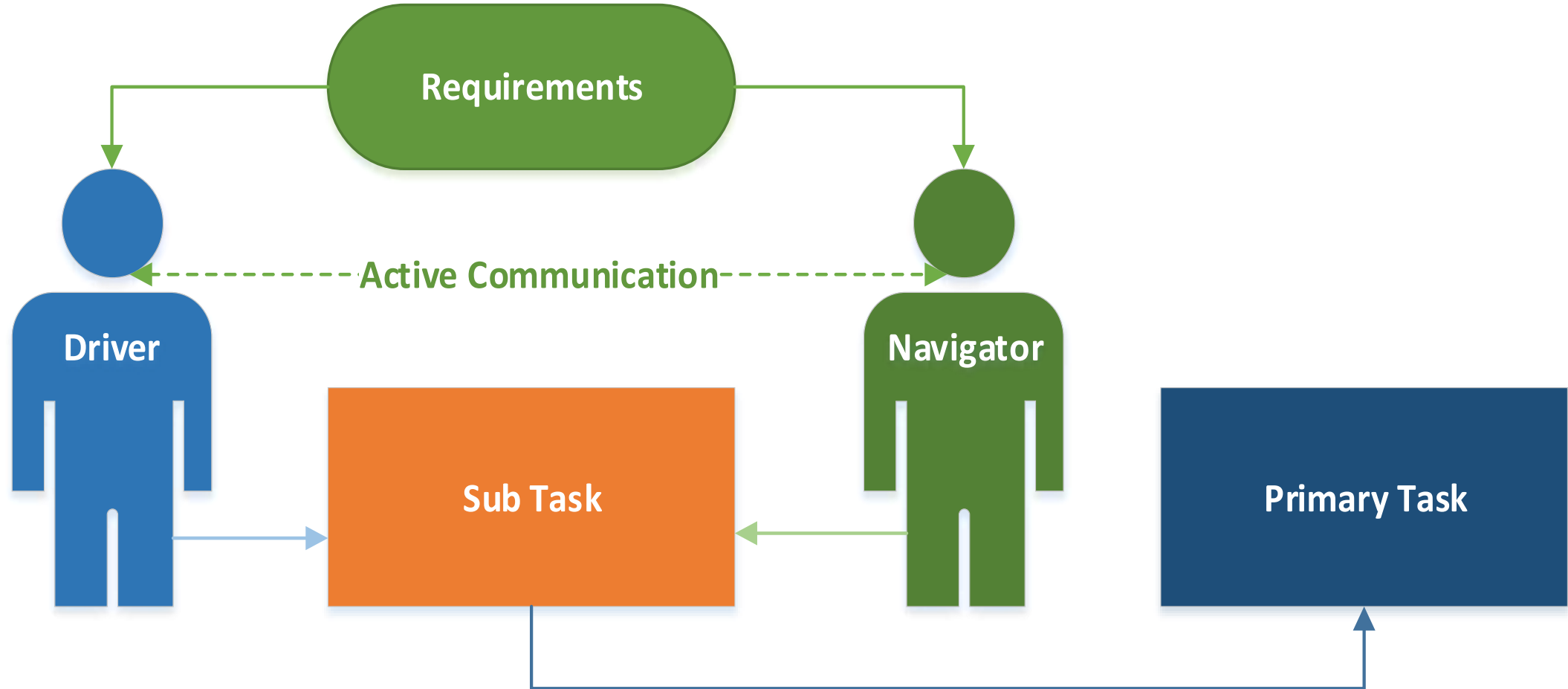
Async State



Dual-Async State



Async Paired State



Pairing Environment Setup



Basic Environment for Pairing




Basic Environment for Each Asynchronous Pairing Partner



Take A Ways

(Pairing \neq )

(Pairing $\equiv \equiv$)

A mallet with a black head and a wooden handle with a red band.

(typeof(Programming) == "Work")

(Pair Programming == Pair Working)



The speed of **WORK** is
limited by the speed of thought



Pairing is not just for Programming



WHEN YOU DON'T PAIR

It makes pandas sad

Todd Merritt



Email: TLMerritt@Gmail.com.com

Twitter: @GeekInterface

LinkedIn:

<https://www.linkedin.com/in/tlmerritt/>

- Over 16 years of Development Experience
- Over 6 years Pair Programming Experience
- Worked with Small Startups to fortune 500 companies
- Interests:
 - App Design/Development
 - Database Development
 - DevOps – Sometimes