

copilot-interaction-models

- [Copilot Interaction Models](#)
 - [Interaction Models Overview](#)
 - [Mode Comparison](#)
 - [Decision Tree: Which Mode to Use?](#)
 - [Interaction Characteristics](#)
 - [Usage Patterns](#)
 - [Ask Mode Flow](#)
 - [Edit Mode Flow](#)
 - [Agent Mode Flow](#)
 - [Key Principle](#)
 - [When Agent Mode Shines](#)
 - [When Agent Mode Is Overkill](#)
 - [See Also](#)

Copilot Interaction Models

This diagram illustrates the three primary interaction models in GitHub Copilot and when to use each.

Interaction Models Overview

```
graph TB
    User[Developer] --> Decision{What do I need?}

    Decision -->|Information/Learning| Ask[Ask Mode]
    Decision -->|Localized Changes| Edit[Edit Mode]
    Decision -->|Multi-step Workflow| Agent[Agent Mode]

    Ask --> AskResult[Explanations<br/>Guidance<br/>No changes]
    Edit --> EditResult[Direct file edits<br/>Scoped modifications]
    Agent --> AgentResult[Planned changes<br/>Human checkpoints<br/>Repository analysis]

    style Ask fill:#e1f5e1
    style Edit fill:#fff4e1
    style Agent fill:#e1e8ff
```

Mode Comparison

```
graph LR
    subgraph "Ask Mode"
        A1[Question] --> A2[Analysis]
        A2 --> A3[Answer]
        A3 -. "No Changes" .-> A4[Learn & Decide]
    end

    subgraph "Edit Mode"
        E1[Instruction] --> E2[Generate Code]
```

```

    E2 --> E3[Apply Changes]
    E3 --> E4[Modified Files]
end

subgraph "Agent Mode"
    G1[Request] --> G2[Analyze Context]
    G2 --> G3[Plan Steps]
    G3 --> G4{Human Review}
    G4 -->|Approve| G5[Execute Step]
    G4 -->|Reject| G6[Revise Plan]
    G5 --> G7{More Steps?}
    G7 -->|Yes| G4
    G7 -->|No| G8[Complete]
end

```

Decision Tree: Which Mode to Use?

```

flowchart TD
    Start([Start([Need Copilot Help])]) --> Q1{Q1{Do I need<br/>code changes?}}

    Q1 -->|No| UseAsk[Use Ask Mode]
    Q1 -->|Yes| Q2{Q2{How many<br/>files affected?}}

    Q2 -->|1-2 files| Q3{Q3{Do I know exactly<br/>what to change?}}
    Q2 -->|3+ files| Q4{Q4{Complex analysis<br/>required?}}

    Q3 -->|Yes| UseEdit[Use Edit Mode]
    Q3 -->|No| UseAgent1[Use Agent Mode]

    Q4 -->|Yes| UseAgent2[Use Agent Mode]
    Q4 -->|No| UseEdit2[Use Edit Mode<br/>with multiple runs]

    UseAsk --> AskEx[Examples:<br/>- Explain pattern<br/>- How does X work?<br/>- Best practices]
    UseEdit --> EditEx[Examples:<br/>- Refactor method<br/>- Add property<br/>- Fix bug in file]
    UseAgent1 --> AgentEx[Examples:<br/>- Multi-file refactor<br/>- Add feature across layers<br/>- Architectural changes]
    UseAgent2 --> AgentEx
    UseEdit2 --> EditEx

    style UseAsk fill:#e1f5e1
    style UseEdit fill:#fff4e1
    style UseEdit2 fill:#fff4e1
    style UseAgent1 fill:#e1e8ff
    style UseAgent2 fill:#e1e8ff

```

Interaction Characteristics

Characteristic	Ask Mode	Edit Mode	Agent Mode
Primary Purpose	Learn & Understand	Modify Code	Complex Workflows
Output Type	Explanations	Code Changes	Planned Steps + Changes
Scope	Informational	1-3 files	Repository-wide

Characteristic	Ask Mode	Edit Mode	Agent Mode
Control Level	High (no changes)	Medium (review edits)	High (checkpoint approvals)
Speed	Fastest	Fast	Deliberate
Context Awareness	Current view	File-focused	Repository-wide
Human Involvement	Question → Answer	Review → Accept/Reject	Approve each step
Undo Complexity	N/A	Simple (revert)	Depends on progress
Best For	Exploration	Targeted fixes	Strategic changes

Usage Patterns

Ask Mode Flow

```
sequenceDiagram
    participant D as Developer
    participant C as Copilot

    D->>C: Question about code/pattern
    C->>C: Analyze context
    C->>D: Explanation/guidance
    D->>D: Make informed decision
    Note over D: No automatic changes
```

Edit Mode Flow

```
sequenceDiagram
    participant D as Developer
    participant C as Copilot
    participant F as Files

    D->>C: Instruction to modify
    C->>F: Read current content
    C->>C: Generate changes
    C->>D: Show diff/preview
    D->>C: Accept or modify
    C->>F: Apply changes
```

Agent Mode Flow

```
sequenceDiagram
    participant D as Developer
    participant A as Agent
    participant R as Repository

    D->>A: Complex request
    A->>R: Analyze codebase
    A->>A: Create execution plan
    A->>D: Present plan
    D->>A: Approve/reject

    loop For each step
        A->>R: Execute step
    end
```

```
A->>D: Show results
D->>A: Checkpoint approval
end

A->>D: Complete workflow
```

Key Principle

Agent Mode is not "better chat"

It's a fundamentally different execution model designed for multi-step, repository-level workflows with human oversight at each critical decision point.

When Agent Mode Shines

- ✅ Multi-file refactoring
- ✅ Adding features across architectural layers
- ✅ Complex analysis requiring repository context
- ✅ Workflows needing plan-execute-review cycles
- ✅ Changes with dependencies and ordering

When Agent Mode Is Overkill

- ❌ Simple file edits
 - ❌ Quick fixes
 - ❌ Single-purpose modifications
 - ❌ Exploratory questions
 - ❌ Learning or understanding code
-

See Also

- [Lab 05: Interaction Models](#)
- [Agent Workflow Patterns](#)
- [Agent vs Instructions vs Prompts](#)