Unified Copilot experience (WIP)

Resources

Design Spec (Figma / Redlines)	
VSO work item	Feature 1855824 [Unified Copilot] Unified Copilot integration across the Shell
Dev Spec	
Related Specs	Spec - Fabric Blueprint Development.docx
User studies	Feedback Supporting a Unified, Persistent Copilot in Fabric – Compilation of user feedback highlighting pain points and need for unified experience. SWIFT Cross Fabric insights AI.pptx

User experience of conversational AI

In traditional systems, user experience is predictable and fixed user flow with predetermined outcomes. AI is shifting user experience from static, predictable flows to dynamic, adaptive experiences that:

Natural language first- we move from button and form interfaces to **conversation**, where users express intent in plain language.

Modular flexibility- design and build components to adapt to varying AI outputs, not assume a single fixed result.

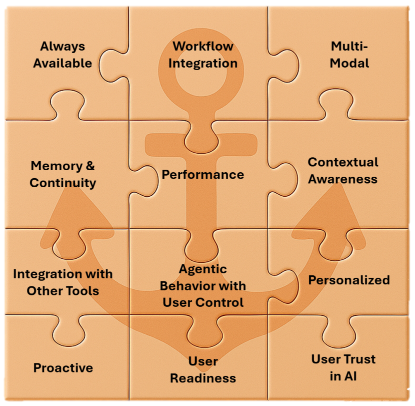
Orchestration over interaction-users become conductors, orchestrating background agents and workflows rather than micromanaging every step.

Crucially, **uncertainty becomes a feature-**a user surface to manage and leverage. And it also has 4 core principles for AI user experience in industry [1]:

- Transparency & Explainability → Make uncertainty visible and understandable. Users need to see **what the system knows and doesn’t know**, and **why** it suggested a particular result.

- User Control & Agency → Let users steer through uncertainty, let users refine, edit, or override recommendations, keep control with the user.
- Design for Graceful Failure → Fail safely when uncertainty is high. Show **clear error states**, helpful **retry options**, and **manual fallbacks**. And provide **guardrails** (constraints, standards) that limit risky actions when confidence is low.
- Focus on capabilities →The goal is **useful, accurate, verifiable and actionable outputs**

In the study of Copilot Chat as a Productivity Anchor [2], they evaluated 9 compete products and talked to 15 commercial genAI users to understand the current landscape of productivity tools anchored in AI and expectations for AI chat. It pointed out that *"While participants see AI as a valuable assistant, they don't yet perceive gen AI tools as a center of productivity due to concerns about trust, reliability, limitations, and task boundaries*." It also highlights that - To make Copilot an anchor for productivity, all these puzzle pieces must come together: reliable performance, agentic capabilities, seamless integration, proactive and personalized experiences, persistent memory, and strong user trust. Only then will users be able to rely on Copilot as the central hub for their daily work.



All of these puzzle pieces are fundamentally tied to the user experience of AI chat. Delivering a great user experience is essential for unlocking the full productivity benefits that AI can offer.

Our Current Situation and Problems

At present, each workload operates with its own Copilot experience, functioning in silos systems.

- In Power BI, Copilot provides support for both semantic models and reports. It assists users in suggesting linguistic model synonyms, generating measure descriptions, writing and explaining DAX queries, and answering questions about data (preview). For reports, Copilot can suggest report pages and visuals (preview), summarize data in a narrative visual, explain report pages or visuals, and is integrated with the Power BI home experience. (Copilot MAU: 349,911)

- For Data Engineering, Copilot is integrated within Notebooks, providing generation of code or markdown for a notebook, adding comments, fixing error or debugging notebook code, analyzing and visualizing data and explaining notebook content. (Copilot MAU: 5,169; MAU: 209,006)

- In Data Factory, Copilot provides support for dataflow gen2 and Pipelines. It assists users in generating new queries for dataflow gen2, creating and running pipelines, summarizing pipeline activity, and troubleshooting pipeline errors. (Copilot MAU: 8,928; DI MAU: 174,748)

- For Data Warehousing, the Copilot is accessible through the SQL query in data warehouse, offering generation of SQL queries, suggesting SQL code completions, fixing code in SQL queries and explaining code in SQL queries. (Copilot MAU: 12,367; MAU: 314,380)

- For SQL database, the Copilot is accessible through the SQL query, offering generation of SQL queries, suggesting SQL code completions, fixing code in SQL queries and explaining code in SQL queries. (Copilot MAU: 2,273; MAU: 10,256)

- For real-time intelligence, the Copilot is accessible through the KQL query and real-time dashboard, enabling users to generate and modify KQL queries as well as generate dashboards (Copilot MAU: 1,469; MAU: 101,380)

There are several challenges:

Experience Inconsistency: Users experience inconsistent interface and interactions with Copilot across the platform. This inconsistency in feature presentation not only causes confusion but also leads disrupting the user's workflow and efficiency.

Duplication of effort: From engineering perspective, this item specific chat experience increases development costs. It also complicates scaling, introduces technical debt, and slows item teams down when they need to react quickly to new AI requirements.

Vibing driven development only: The current Copilot chat experience design is for the vibing development approach, which is freeform and open-ended, and it has been beneficial for early-stage flexibility. With the blueprint is as new creation experience for spec-
driving development in Fabric, the existing Copilot chat experience falls short. It lacks the structured

guidance and rigor required for the spec-driven development, which needs guided, structured UX and interactions between UI elements.

Gaps in multi-agent support: Most of the items Copilot chat experience are designed around a single dedicated AI agent for the specific item, and without multiple agents support for specialized capabilities, which would allow users to seamlessly collaborate with domain-specific agents within the same workflow.

Limited workflow integration: The chat experience functions primarily as an isolated conversational interface for specific items, rather than as a deeply integrated component of the user's end-to-end workflow. It limits user actionability—users cannot view and interact on multiple items or UI elements while work with Copilot, preventing them from following along within the workflow.

Context reset and limited continuity: lacks of user conversational history and context. When a user switches items, the Copilot session loses the prior context and conversation. Users must restate the context and help Copilot remember what was done.

Gaps in multi-modal support & contextual awareness: Does not support multi-modal interactions. Inadequate support for various interaction methods and for capturing contextual information

Incomplete personalization: Users receive experiences tailored only to specific items, rather than personalized holistic support spanning their entire workflow.

These challenges underscore the urgent need for a unified and intelligent Copilot chat and workflow interaction experience—one that delivers seamless, consistent, and comprehensive support across all workflows. Addressing these gaps will not only elevate the user experience but also unlock the full productivity benefits that AI can offer

Goals

Goals	Priority
Provide consistent Copilot chat and UI elements interactions	P0
Faster workflows- Persistent context and unified chat	P0

Commented [BS1]: These feel potentially redundant. Perhaps we align that our primary goal is "One Copilot" and that the success metrics we expect to see are things like faster workflows, lower frustration scores, etc.

Non- Goals

Non-goals	Reason

The capability to build data project with Copilot	This feature does not aim to provide the functionality of building data solution.
Unified AI stacks and orchestration	This is the backend of the unified Copilot, PBI team + ILDC team

Success Metrics

No.	Outcome	Measure	Target
1	The unified Copilot experience empowers users to interact with a single, consistent Copilot interface throughout Fabric.	Reduction in time spent switching between items or manual context setup	
2	Minimizes navigational confusion, eliminates context resets, and supports multi-item operation for faster workflow	Workflow Success Rate Time from initiation to completion	
3	User Satisfaction	Feedback on the unified Copilot experience.	

User scenarios

Hero scenario for M0

Ash is working on a marketing trends analysis request. With unified creator experience now available in Fabric, Ash plans to use it to build a report.

Open Unified Copilot and start with suggested prompt

Ash can open the unified Copilot chat interface. Ash sees the prompt Gallery, which contains system predefined prompt for help users to discover capabilities. And Ash notices the prompt for the blueprint, and Ash decides to start from there to build a blueprint, in which the detail requirement is specified. Ash selects the creator agent.

Context binding implicitly

Ash is in My workspace and sees My workspace is bound as context in Copilot by default.

Commented [JV2]: I do not believe this is a problem that all our user personas have. In particular, the business user (Nat) persona targeted by the critically important "Chat With Your Data" (CWYD) scenario is not a creator / developer. They consume existing content and ask data questions of that content (currently Power BI reports and semantic models and Data Agents. Although, I'm sure other workloads, notably RTI, have some consumer facing artifacts as well).

This is a large and critically important population of users. In our work on CWYD so far, these users don't understand workspaces, capacities, or even most item types, even semantic models. In fact, we're making changes to the CWYD experience in response to feedback that these users are not comfortable picking from a list of items in order to get the answer to a data question.

These are very different users from the developer/creator personas targeted by most of Fabric, and I believe we must be very careful not to disrupt their experience even as we improve things for other Fabric user personas.

Commented [BS2R2]: Makes sense, John. I think the intended scope of this work is target at the Fabric Shell-- and in turn the creator/developer audience.

As a thought exercise... do we imagine that completely separating the CWYD from the rest of the Unified Co ... [1]

Commented [BS2R3]: @Drew Voegele I'm curious on your thoughts here as well, as it's sort of an extension of a thread us 3 had last week in planning.

Commented [DV2R4]: Well, it seems like it is the user prompt itself that informs the orchestrator regarding which other agent to pull in whether that's CWYD or any other creator agent. If in Power BI, if the user is prompting to achieve a creator outcome that may be better done in Fabric, this could be a good opportunity to start moving them over.

Commented [BS2R5]: Makes sense to me, Drew. I think at this stage it's valuable for us to explore some early mocks of how this might manifest. It should help us answer some of the concerns @John Vulner has around: Do we have different entrypoints/modes for CWYD vs. Creator mode? Do we depend on our agent's ability to effectively triage a prompt into a CWYD response or a creator ... [2]

Commented [BS2R6]: Also, @Dan Liu I think we should change the work "workloads" --- perhaps "Fabric items" is more appropriate for now?

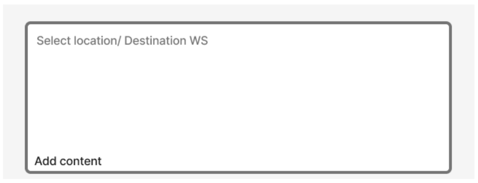
Commented [BS3]: @Dan Liu I think we should be careful in these early stages to avoid creating too many dependencies between the Blueprint concept and this unified concept.

@Rui Romano is also outlining the base requirements for the Blueprint, and therefore should help digest the U ... [3]

Ash asks Copilot to create a new workspace to have the related new items saved into the new workspace. Workspace is created and displayed on the side page. The context is automatically set to the new workspace.

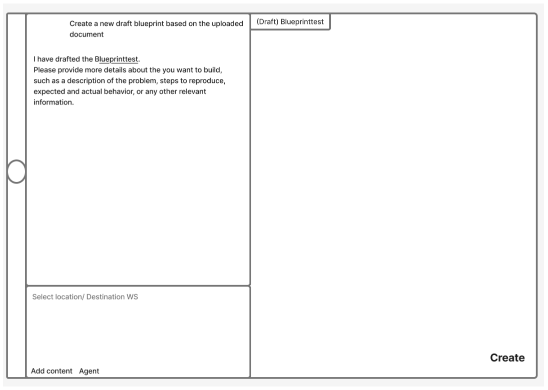
To create blueprint, Ash put the message in chat box as:

Ash: Create a new draft blueprint based on the uploaded document



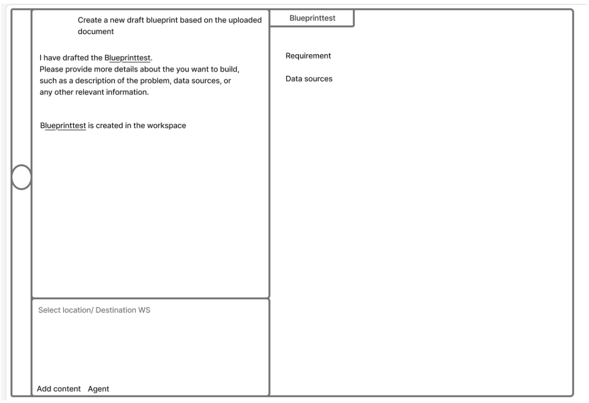
Preview and create item

Copilot drafts the blueprint, and shows the drafted blueprint in the side pane for preview. The interface transitions from a full-screen layout to a split-view layout for surfacing items drafted by Copilot. And Ash can resize the chat pane.


Create a new draft blueprint based on the uploaded document

I have drafted the Blueprinttest.
Please provide more details about the you want to build, such as a description of the problem, steps to reproduce, expected and actual behavior, or any other relevant information.

Ash reviews the draft content in the blueprint and clicks Create button to apply the changes. The blueprint is created.


Create a new draft blueprint based on the uploaded document

I have drafted the Blueprinttest.
Please provide more details about the you want to build, such as a description of the problem, data sources, or any other relevant information.

Blueprinttest is created in the workspace

Update part of item content with vibing development

Ash wants to update a specific paragraph in the blueprint. Ash updates the blueprint by chatting with Copilot.

Ash selects specific parts and puts the message into chat box as:

Ash: Update the requirement to make sure the regional data in datasource 1 is included

Copilot: The blueprint is updated.

Copilot updates the blueprint content based on Ash’s needs. Ash can undo the changes.

Ash keeps finalizing the blueprint with Copilot.

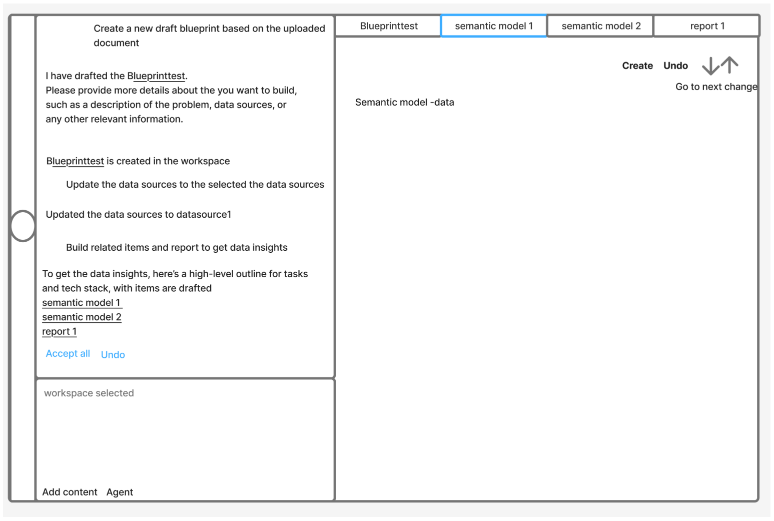
Create multiple items

Ash asks the Copilot to create related items and reports to get data insights. Copilot creates outlines how many tasks and how much time required for each task, and show how many tasks left and the remaining time

Objective	Produce a focused, decision-ready analysis that answers the primary business question with validated metrics, supporting diagnostics, and a fit-for-purpose deliverable (report/semantic model /dashboard).
Status	**IN PROGRESS** 0/9 tasks

Progress	0%
Time remaining	56 min
Current task	Confirming objectives and constraints

And Copilot creates a task flow , 2 semantic models and 1 report in draft state, with all of them showing in the side pane.


Create a new draft blueprint based on the uploaded document

I have drafted the Blueprinttest.
Please provide more details about the you want to build, such as a description of the problem, data sources, or any other relevant information.

Blueprinttest is created in the workspace

Update the data sources to the selected the data sources

Updated the data sources to datasource1

Build related items and report to get data insights

To get the data insights, here's a high-level outline for tasks and tech stack, with items are drafted
semantic model 1
semantic model 2
report 1

Semantic model -data

Ash reviews items and content in the side pane one by one by switching the tabs and decides to Accept all. Ash closes the tabs in the side pane after the reviewing.

Ash clicks link in the chat to reopen semantic model2 in the side pane

Delete items

Ash decides to delete semantic model2, and puts message in chat box as:

Ash: Delete semantic model2

Copilot: are you sure to delete semantic model2

Ash puts the Yes in chat and sends it Copilot to confirm the deletion of item. The semantic model2 is deleted and disappears in the side pane.

Select and switch agent in a conversation

Ash selects Analysis agent in the chat experience to ask questions about the data related to marketing trends.

Key features of the unified chat and UI interaction experience

- **Persistent Copilot chat interface:** makes the chat a **first-class interface** for actions.
  - Prompt gallery with system prompts for common tasks (e.g., "Develop a data solution," "Create a blueprint")-P0;
  - Provide feedback on Copilot output-P0
  - Allow users to start new conversation-P0
  - Select and switch agent in the mid- conversation-P0/P1
  - Resize chat pane -P0
  - Bookmarking and quick access to favorite prompts.-P2
- **Context Selector:**
  - **S**upport the implicit context selection based on users current position, workspace, and the item currently open. -P0
  - Explicitly bind Copilot's context to specific workspaces, folders, items, or parts of the item content with visual indication of current context; -P1
  - option to upload documents for Copilot to reference.-P1
- **Live update in side view:** As Copilot drives creation, update, and deletion of items, users can view changes in the side view. -P0
- **Side-by-side interactivity:**
  - Dynamic transitions from a full-screen layout to a split-view layout for surfacing items in the side view-P0
  - UI or item navigation by clicking links in Copilot output-P0
  - Support open multiple items/UIs and easily close them in the side view- P0
- **Actionable Response Controls:** Users can take actions directly from Copilot responses
  - Open item-P0
  - Accept/confirm all changes-P0
  - Undo-P0
  - Provide fine-grained control over what they accept or undo -P1
- **Robust Error Handling and Edge Case Management:** handle unexpected situations (long run request, network errors), failures, and unusual user actions (message too large), ensuring a smooth and reliable user experience-P0
- **Change Tracking and Diff Viewer:** show changes (additions, deletions, modifications) made to items, allowing users to review, accept, or undo updates.-P1

Commented [BS4]: There's a second pivot on this I think we should capture.

You've mentioned *explicit* context selection, which is great and critically important.

I believe we should also scope *implicit* context selection--meaning if I have 4 items open as tabs, my prompt should be contextualized against those open items/workspaces as well.

The scenario here that many workloads will want before they integrate is, for example:
--- User opens a Dataflow Gen2, and opens the unified Copilot pane.
--- User prompts: "Let's ingest some data from [selected excel] as a source."
--- Copilot knows that the ingestion should take place in the Dataflow Gen2 that is **open**, not another one or a new one.

At the very least, perhaps we would create a mechanism for 'quick selection' of items that are open to be added as context for a prompt.

Commented [DL4R2]: Thanks for calling this out! Absolutely right—implicit context selection is critical. Captured this as part of the scope.

- **Chat History and Thread Management:** continue conversations across sessions and items, with Copilot retaining the thread and context.-P1
- **Model selector :**Lets users pick the most appropriate model or agent for their current context or task **–P1**
- **Guided Follow-Ups:** Offer the next question or next action after a response to help users to bridge the response with next step/action.-P2
- **Personalized**: AI assistance feels like it's tailored to the user's unique role, needs, and workflows.-P2

Milestones (WIP)

Mil.	Goals
M0	Deliver the base unified chat infrastructure and seamlessly integrated across workflows, enabling users to interact and take action within a consistent Copilot experience
M1	Deeply integrate Copilot into users' unique workflows by enabling explicit context binding, model and agent selection, chat continuity, and fine-grained control over actions.
M2	Make Copilot experiences personalized and embedded in users' specific workflows.

I cans (WIP)

Pri.	**I cans**	**Additional details/consideration**	**Milestones**
		Copilot	
P1	I can browse the prompt gallery to see all available capabilities at a glance.	Workload team can configure and add their prompts into gallery	
P1	I can bookmark prompts for quick access in the future	Quick access to favorite tasks, making it easy to discover and start common workflows.	

P2	I can open and rely on unified Copilot to at any page in Fabric	Copilot remains available while editing items with “copilot by your side” experience everywhere in Fabric	
P1	I can pick the most appropriate models for the conversation	Each conversation can only configure one model	
P1	I can select the agent for the conversation		
P2	I can use the unified Copilot directly from the home page and start a task without first launching an item.	Shift the home page from discovery-only to Copilot-centric, so users can start with goals	
Context Selector -Provide/Bind Context to Copilot			
P0	I can rely on unified Copilot to automatically understand the context of what I’m working on—such as my current workspace, item.	Users don’t have to manually specify it when asking for help or performing actions	
P1	I can select a workspace to bind Copilot’s context to the workspace	Ensures all responses and changes (draft items, outputs) are scoped to a specific workspace like Workspace A	
P2	I can select multiple workspaces to bind Copilot’s context to the workspaces	For cross workspace project, users can provide Copilot access to more than one relevant workspace	
P0	I can select folders to bind Copilot’s context to folders	Narrow down content scope within a Folder	
P0	I can select items to bind Copilot’s context to items	Focus responses on specific items	
P0	I can upload files to provide context to Copilot		
P1	I can select parts of an item’s content as context for Copilot	Highlight certain sections for Copilot to revise or regenerate, update.	

P0	I can see the selected context in the chat box	Visual verification helps the user confirm which context Copilot will use	
P1	I can revise the prompt that was sent in order to generate a new response.	Adjust tone, scope, or detail in follow-ups without starting over	
Item(s) Navigation and Navigate Between Changes			
P0	I can open navigate to the item(s) from the Copilot interface in the side page	Click item links in chat to open them in the Fabric side pane for immediate review.	
P0	I can view the draft items created by Copilot in the side page	Copilot generates items based on my prompts, displays changes in the side page	
P1	I can navigate across multiple changed items to see what Copilot proposed.	Quickly step through different updates (e.g., 1 blueprint + 6 new items) with a built-in navigation bar.	
P1	I can jump directly to a changed section within an item	Save time by scrolling to highlighted revised sections inside items.	
Change Tracking and Diff Viewer			
P0	I can view overviews of item changes	e.g. 6 items added; 1 item updated; Identify which items were added, updated, or deleted	
P1	I can see highlights of changes within each item to understand what's new, updated, or deleted.	e.g. 2 lines deleted in itemX, Color-coded differences (added = green, removed = red) for fast scanning of edits	
P2	I can open the Diff Viewer to compare original content with Copilot's updated version side by side.	Full traceability for sensitive edits before applying them, similar to code review workflows.	
Actionable Response Controls (Preview-Decide-Apply-undo-improve)			

P0	I can accept Copilot’s response completely to apply all suggested changes.	Apply all of Copilot’s suggested changes in one click	
P0	I can accept or reject individual changes selectively, such as specific sections or selected items.	Pick specific changes like blueprint edits while ignoring report changes.	
P0	I can undo a specific Copilot action immediately after it is applied.	Quick rollback option after applying incorrect updates	
P0	I can revert changes on a particular item while keeping updates on other items.	Granular control for selective rollback in multi-item workflows	
P1	I can insert Copilot’s response into a specific position in the item when drafting or editing.	AI inserts content inline without overwriting existing text	
P0	I can copy results, or provide feedback to the response		
P0	I can retry an action if the initial Copilot response does not align with requirements.	Trigger regeneration immediately with modified instructions.	
P0	I can confirm before performing high-impact actions, such as deleting an item, to prevent accidental loss	Ensuring critical actions require explicit approval.	
Chat History and Thread Management:			
P1	I can continue conversations with Copilot across sessions	Persistent personal history allows users to resume without re-uploading context the next day	
P1	I can select conversation and delete the conversation	Control over data and privacy by removing chat logs that are no longer needed.	
Error Handling and Edge Case Management			

P0	I can identify when Copilot cannot perform a requested action		
P0	I can read and interpret Copilot’s error messages to understand what went wrong		

Appendix: Relevant Documents

Open questions:

References

1) Medium, UX for AI Products, 2024; Moze Studio, UX design for AI guide, 2024; Raw.Studio, UX Design For Generative AI, 2024; LinkedIn, UI/UX Design for AI Products, 2024; Nielsen Norman Group, Prioritize Smarts over Sentience, 2024
2) Copilot Chat as a Productivity Anchor | Assembling the Anchored AI Puzzle: How to Make Copilot Chat in WXP a Center for Productivity 🧩; Assembling the Anchored AI Puzzle How to Make Copilot Chat a Center for Productivity.pptx

Page 5: [1] Commented [BS2R2] Brent Sandifer 10/28/25 11:22:00 AM

Makes sense, John. I think the intended scope of this work is target at the Fabric Shell-- and in turn the creator/developer audience.

As a thought exercise... do we imagine that completely separating the CWYD from the rest of the Unified Copilot is an appropriate response to your concern? I could imagine a world where...

1. When users are in the PBI portal (using the switcher) they have ONLY the CWYD agents in the immersive.

2. When users are in the Fabric portal (using the switcher) they have ONLY the unified creator/developer stack.

Again, just a thought exercise... but curious to hear your thoughts on if something like this appropriately addresses your concerns AND delivers the right value to creators.

Page 5: [2] Commented [BS2R5] Brent Sandifer 10/29/25 11:55:00 AM

Makes sense to me, Drew. I think at this stage it's valuable for us to explore some early mocks of how this might manifest. It should help us answer some of the concerns @John Vulner has around: Do we have different entrypoints/modes for CWYD vs. Creator mode? Do we depend on our agent's ability to effectively triage a prompt into a CWYD response or a creator response? What level of explicit vs implicit control does a user have to influence this?

I will start some discussions with my team on this soon.

Page 5: [3] Commented [BS3] Brent Sandifer 10/28/25 11:27:00 AM

@Dan Liu I think we should be careful in these early stages to avoid creating too many dependencies between the Blueprint concept and this unified concept.

@Rui Romano is also outlining the base requirements for the Blueprint, and therefore should help digest the Unified Copilot plan we establish here to help land the Blueprint interactions.

In the fullness of time, the Unified Copilot will interact with all other items/agents in the system---including the blueprint.