SDS PODCAST EPISODE 918: MULTI-AGENT SYSTEMS WITH CREWAI

Jon Krohn: 00:00 This is episode number 918 on Multi-Agent Systems with CrewAI. Welcome back to the SuperDataScience Podcast. I am your host, Jon Krohn. Today's episode is a crisp overview of multi-agent systems and specifically CrewAI an extremely popular framework for creating and managing multi-agent teams. I'll cover what CrewAI is, how it works, a few concrete use cases, a quick comparison to earlier agent frameworks and why it matters for your workflow. All right, let's jump in. So first off, CrewAI is an open source python framework for composing teams of agents, teams of AI agents that collaborate like a small, well, like a crew or a team or a small company. Instead of asking one general model to do everything, you stand up specialized agents such as one that's a researcher, another that's a planner, another that's a writer, another that's an engineer.

00:55 And each agent on that crew has a clear goal and specific tools that they need for their particular specialization. The crew then coordinates to produce an outcome that's larger than the sum of its parts. In practical terms, CrewAI gives you structure around multi-step multi-agent work so that complex projects become repeatable, auditable and faster. So how does it all work with the CrewAI framework? We think in terms of roles, tasks and handoffs, the roles you assign to each of the agents in your crew, the tasks that get assigned to crew members and the handoffs wherein information flows between different agents on the crew, you define each agent to have a particular role. You break a project into tasks and you let the framework automatically route work between agents with shared memory and critically to make sure these things all work safely with guardrails as well.

01:49 So you might have one agent that gathers facts, another that synthesizes information, another that critiques, and a final agent that assembles the deliverable all in some consecutive workflow that you've defined or potentially

non-sequential. And there's different degrees of autonomy you can define. So you can run the crew with substantial autonomy that's great for things like open-ended research and creativity, or you can constrain the crew to have a more deterministic workflow when you need reliability auditability or some kind of tight software agreement that you have with say, a client of yours. The key is that you have explicit roles and deliberate handoffs, so each agent knows when to stop, what to produce, and who to pass it to. Hopefully the power and flexibility of multi-agent systems is starting to sink in. But here are three different use cases to make it concrete. First software development, something that many of us are familiar with, you could have a crew that can review pull requests automatically.

02:52 So you could have a static analysis agent that flags security and style issues. A testing agent that proposes unit tests, a reviewer agent that explains risks and suggests differences. And humans would still need to approve and merge the pull request, but the crew handles the heavy lifting, accelerating code quality while reducing toil. As a second example, think about content creation like creating a podcast episode. You could have a research agent that compiles citations, a writer agent that drafts with structure and voice and an editor agent that revises for clarity, accuracy, and tone. Because each role is persistent, you get consistent behavior across runs. Your writer keeps the same brand voice, your editor enforces the same style guide, so outputs improve over time. The result is publication ready copy with fewer cycles. Again. However, just like with the software example, you are probably going to want humans in the loop for the foreseeable future on these outputs to ensure that it really is publication ready copy.

03:59 And I've got some more gotchas for this end of the episode for things that you need to be looking out for in this

multi-agent system world. But before we get there, here's a third and final example for you. So first we had software development. Second, we had content creation. A third example is here in industrial operations. So imagine a supply chain or a customer support scenario. You could have a watchtower agent that monitors signals inventory, whether social chatter, a planner agent that recalculates options, and a negotiator or comms agent that reaches out to vendors or customers with proposed adjustments. When an exception hits, the crew reacts in minutes instead of hours, and every decision is logged for audit and learning something that can be trickier with people. So how does CrewAI compare to earlier agent approaches? Traditional single agent plus tools setups are powerful for bounded, well constrained tasks, but they tend to blur roles and require constant prompt juggling to maintain context.

05:00 Early multi-agent experiments proved the idea, yet often lacked stability. So they proved the idea of having something like crew but didn't have the stability. So CrewAI leans into both specialization and coordination. You get durable roles, explicit task decomposition and structured handoffs. You get the creativity of autonomous agents with the governance of a defined process. In practice, that means fewer loops to get things right, less brittle prompting and easier scaling from one-off experiments to production workflows for data scientists, AI engineers, software developers, or any other practitioners looking to build multi-agent systems. True AI provides a straightforward architecture for going forward. You define agents in code, so things like name, role, goals, allowed tools, and any safety constraints. You define tasks including acceptance criteria and expected artifacts. Then you choose how work flows. You could have a freeform crew for exploration, a stricter flow for determinism or a hybrid that uses a flow to call a crew at key steps.

06:07 Logging and intermediate artifacts make runs inspectable, which is essential for debugging and for regulated environments. And the good news, more good news is that swapping models or tools is a simple configuration change, not a rewrite. So you can evolve the system quickly as requirements change. Now, the bigger picture, multi-agent workflows shift AI from a clever assistant to an actual team member or like a team of team members that can unlock step function gains in productivity. Projects that used to require multiple expert humans in the loop at each phase of the project can now be initiated with a single well scoped brief. It also elevates human work when crews of agents handle the rote grind, collecting, summarizing, formatting. We humans get to spend more time on judgment, taste, and strategy with great new power. Of course also comes new responsibilities. This is what I was talking about earlier, the things to look out for.

07:03 So some of the key ones when running a multi-agent system include defining review gates, tracking sources, restricting tool permissions, monitoring spend, and keeping a human in the loop for consequential decisions. To wrap with those caveats and concerns aside, things to look out for CrewAI's core idea is simple and potent. Specialized agents coordinate them well and let them work together to toward a shared objective. If you've dabbled with single agent prompts and hit limits on their capabilities, accrue may be the next logical step for you. Start small codify roles. You already play decompose one weekly task into two or three agent handoffs and then iterate from there. The payoff is compounding. You get cleaner processes, faster cycles and results that feel like a competent team delivered them. And as I've mentioned many times on this podcast before, if you're not sure where to start with multi-agent systems in your organization or maybe even in your personal life, your favorite conversational agent, be it Chat, GBT, Claude or

Gemini is only a browser tap away and can help you ideate on where to get started.

08:08 This may all have sounded like a long ad for CrewAI , but they have in no way sponsored me or this show. I'm simply a big fan. Indeed. If you want to learn more about CrewAI and engineering teams of AI agents, you can check out the four hour workshop that I published on YouTube. It's all available for free. And I did it with my brilliant longtime friend, ed Donner. We've got a link to that in the show notes for you. And of course we have the GitHub, URL for the open source CrewAI repo in the show notes for you as well. Alright, that's it for today's episode. I'm Jon Krohn and you've been listening to this SuperDataScience podcast. If you enjoyed today's episode or know someone who might consider sharing this episode with them, leave a review of the show on your favorite podcasting platform. Tag me in a LinkedIn post with your thoughts, and if you aren't already, be sure to subscribe to the show. Most importantly, however, we hope you'll just keep on listening. Until next time, keep on rocking it out there, and I'm looking forward to enjoying another round of the SuperDataScience podcast with you very soon.