**EOP/ECM education agent**

**Preface**

From the technical perspective of building the AI agent, the slides Virginia attached in the email thread on August 25 were already detailed and well-reasoned. Yet, fully implementing the contents within them may not be particularly feasible in the coming month or two — at least that’s my understanding. It might be worth considering developing a separate *Brief Communication* for *Nature Methods*, *Nature Machine Intelligence*, or *Nature Computational Science* once the agent is fully constructed. At this stage, the version intended for our *Policy Forum* manuscript should serve as a basic demo.

**Current aim**

At present, our demo mainly aims to address the concerns raised by the editors. Building the ECM poses an unavoidable barrier, especially for researchers who lack software engineering expertise. If we cannot substantially lower this entry barrier through educational means, the broader adoption of the initiative will face significant challenges. To address this concern, our agent demo may need to include the following preparations, from users’ perspective.

1. Explaining ECM’s significance

We aim for the agent to articulate the advantages of the ECM from multiple perspectives, explain its broader significance, and foster wider consensus through interactive engagement with users. Beyond the foundational concepts introduced in our Policy Forum, our goal is for the agent to draw on the disciplinary background or knowledge system of users, using analogies and context-aware explanations, to help them intuitively grasp the value and implications of the ECM within their respective fields. Again, the essence of equipping the agent with this capability is to maximize outreach and foster consensus without requiring significant additional human effort.

* **User (biologist):** I understand that the ECM organizes research software evidence, but how is this relevant to my work in molecular biology?
* **Agent:** Think of the ECM as a form of experimental protocol for computation. Just as your lab records each reagent, condition, and step to make an experiment reproducible, the ECM captures the computational equivalents, including datasets, code, environment, parameters, and outputs, so that others can verify and build on your findings.

2. ECM-guided software development prompting

To complement the diagnostic analysis, the agent can assist researchers during software development. When a user is building a research repository without a clear understanding of what needs to be disclosed, the agent provides evidence-oriented development hints. Based on the ECM structure (i.e., basic evidence chain), the agent dynamically suggests the next evidentiary elements to include (e.g., code lines, scripts, configuration files, metadata, or documentation). The user can select among proposed options (a, b, c), after which the agent generates corresponding templates or code snippets to fill disclosure gaps. This interaction enables researchers, especially those without formal software-engineering expertise, to construct research software that is both functional and evidentially complete.

* **User:** I’ve uploaded my analysis script, but I’m not sure what else I need to include for transparency.
* **Agent:** Your repository currently contains analysis.py and results.csv, but lacks environment and execution records. Suggested next steps: (a) Generate a requirements.txt to lock dependencies. (b) Create a run.sh script showing how results were produced. (c) Add a provenance record linking analysis.py → results.csv.
* **User:** Let’s do b and c.
* **Agent:** Done. I’ve created run.sh with the command …

3. ECM-ization of established research software

Researchers may use the agent after the research software has been fully developed. At that stage, the agent can evaluate its evidentiary completeness by transforming the repository into an evidence chain structure. Specifically, the agent automatically identifies all computational artifacts — data files, scripts, configurations, and outputs — and maps their logical relationships. It then detects where the evidence chain is broken or incomplete, such as missing scripts, undefined data sources, or absent execution records.

The output takes the form of interactive, evidence-based suggestions: the agent provides structured feedback on disclosure gaps and proposes specific corrective actions. This allows researchers to clearly see how their software aligns with the ECM and what additional elements are required to make their results fully verifiable.

4. Reorganizing fragmented scripts based on the ECM

In some cases, researchers already possess a collection of scattered scripts rather than a well-structured research repository. When this occurs, the agent can help reorganize the scripts into a coherent, evidence-oriented code repository. Drawing on both the content of the scripts and the researchers’ description of the overall experimental pipelines, the agent infers dependencies among files, reconstructs the logical order of execution, and groups related components under the appropriate stages of the evidence chain. Through this process, unstructured code is transformed into a transparent and reproducible framework aligned with the principles of the ECM.

5. Field-specific disclosure guidance

TODO, will discuss in the meeting

6. Suggestions for alternative solutions

TODO, will discuss in the meeting