
AI and the Developer

John M. Daughtry III

The Applied Research Laboratory
and The College of Information
Sciences and Technology
The Pennsylvania State University
210 ARL Building
University Park, PA 16802 USA
daughtry@psu.edu

Abstract

The field of artificial intelligence (AI) produces tools and techniques that seek to make machines more intelligent. There are two users of these artifacts. The most obvious user is the end-user. However, before an end-user has a chance to interact with the system, a developer has to design a system that makes use of AI for that user. My position in this paper is that we need to spend more time studying many AI tools and techniques with respect to the first user to interact with it, the developer.

Issues Facing Developers who Use AI

As an engineer with the Applied Research Laboratory, I am a professional software engineer. My development experience lies in the realms of multi-sensor data fusion, small business point-of-sale systems, and enterprise information management systems. At the same time, I am a researcher in human-computer interaction with a specific interest in software development. The balance of this paper discusses the developer as a user of AI tools.

Whether an ACT-R model, a latent semantic analysis toolkit, a speech recognition engine, or a neural network, AI tools and techniques pose many problems for developers.

Software developers seek a best-fit solution to a given problem. Towards this end, they have to understand the capabilities and limitations of the technologies employed. A database, for example, may have to be capable of handing some number of inserts per second. However, many AI tools and techniques are harder to understand. For example, they may have to understand what a user means by some block of text. The very nature of AI necessitates that at some level it is unpredictable. Often, developers have to implement a solution in order to know if it will do what they need. Even if you do a trial implementation and it is successful, how can you know if it will work next time under slightly different circumstances? In many cases, you can't.

Another problem that faces developers that use AI is that they must often treat them as a black box tool. AI tools and techniques are very complex. Can an application developer be expected to alter the code in a natural language parser? They could, but the cost would generally be too high for it to be feasible. Therefore, they have to treat the AI tool as a black box, just as they treat an operating system, application server, or database management system.

At a lower level of analysis, developers have to work with an application programming interface (API) to use an AI tool. This API can take many forms, such as a web service, framework, library, or toolkit. If developers have to treat AI as a black box, the API takes on an even higher level of importance.

There has been some research into the use and usability of APIs. For example, Rosson and Carroll [1] found that developers tended to reuse code fragments that made use of an API. A more recent evaluation by Ellis, Stylos, and Myers [2] found that the factory pattern diminished usability in certain instances. Another example is a conceptual map of the API design space recently presented by Stylos and Myers [3].

The essential problem I see in using an AI API is that it is unpredictable. This manifests itself in the use of an AI API. For example, how do you test your use of an unpredictable tool? And, how do you know what to change in your use of an AI API when it isn't working as expected? Changes to the parameters used by AI APIs are also unpredictable.

These problems are related to the problems faced by an end-user of AI. And, with more and more end-users developing software themselves, the boundaries blur.

References

- [1] Rosson, M.B. and Carroll, J.M. The reuse of uses in Smalltalk programming. *ACM Transactions on Computer-Human Interaction*, 3 (3). 219-253.
- [2] Ellis, B., Stylos, J. and Myers, B., The Factory Pattern in API Design: A Usability Evaluation. in Proceedings of the 29th International Conference on Software Engineering, (2007), 302-312.
- [3] Stylos, J. and Myers, B., Mapping the Space of API Design Decisions. in Proceedings of Visual Languages and Human-Centric Computing 2007, (2007).