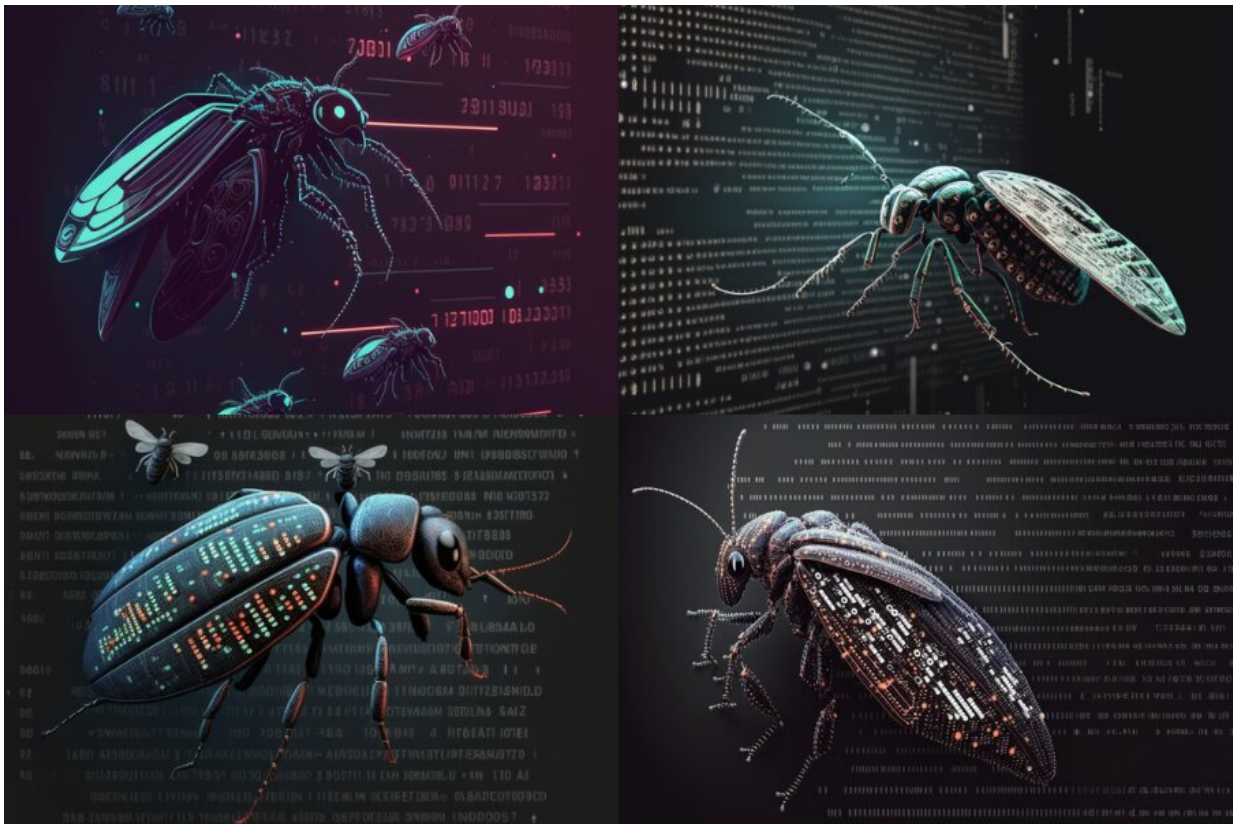
New software tool provides an easier way to debug any domain-specific programming language

April 10 2023, by Steve Nadis



D2X is a domain-specific language debugging infrastructure that works with most standard debuggers without any modifications and is easily extensible to capture all the domain-specific information the end-user cares about. Credit: Alex Shipps/MIT CSAIL via Midjourney

Sometime in 2019, MIT Ph.D. student Ajay Brahmakshatriya formulated a simple, though still quite challenging, goal. He wanted to make it possible for people who had expertise in a particular domain—such as climate modeling, bioinformatics, or architecture—to write their own programming languages, so-called domain-specific languages (or DSLs), even if they had little or no experience in creating programming languages.

A member of the research group headed by MIT Professor Saman Amarasinghe in the Institute's Computer Science and Artificial Intelligence Laboratory (CSAIL), Brahmakshatriya wanted these languages to come with all the auxiliary functions people would need to comfortably utilize them, including tools for debugging. This process for getting rid of errors in a piece of software is essential, he and Amarasinghe agreed, as they have called the lack of debugging support "the Achilles heel for DSLs."

It's been a productive few years for both of them. In 2021, Brahmakshatriya and Amarasinghe introduced BuildIt, a software package that greatly simplifies the task of creating DSLs. And last month, at an international conference in Montreal co-sponsored by the Association for Computing Machinery, the duo introduced D2X, a tool that makes it easy to add debugging to any DSL and has been shown to work particularly well with BuildIt. Their paper on the work even won one of two Distinguished Paper Awards given at the conference.

The main reason for producing a language in a specialized domain, Brahmakshatriya explains, "is to promote ease of use." An image-processing DSL, for example, could have a function that says "blur the entire image." Issuing that same command in a general-purpose language would require many more lines of code, notes Brahmakshatriya. "That's part of the reason to use a DSL. The other is performance." Because the operations are specific to that domain, they can be more readily optimized—carried out in the proper order, and hence completed more efficiently and quickly.

Brahmakshatriya describes BuildIt as "a DSL for creating DSLs." It facilitates a multistep procedure for taking an existing, all-purpose programming language and paring it down until it becomes specialized in just the right way. "Suppose you have a problem, and you want to write a program to solve it," he says. "You could write a program to solve it in its entirety, or you could write a smaller program to solve just the subclass of the problem you're interested in. The more specialized you make the program, the faster it runs." BuildIt is designed to construct DSLs with those guiding principles in mind.

Halide—an image processing language invented in 2012, years before BuildIt was around—is one of the first DSLs to come out of Amarasinghe's group. Its development was led by then-graduate student Jonathan-Ragan Kelley and Andrew Adams, a CSAIL postdoc at the time. "Halide is very popular now, and it is used in many Adobe applications, including Photoshop, but it still doesn't have a debugger," Amarasinghe says. The reason for that, he adds, "is that debuggers are very complicated. It's very hard to write them, which is why most small DSLs don't have debugging support."

That's not a desirable state of affairs, according to Brahmakshatriya, who insists that every DSL should have its own debugger. "You can't directly use existing debuggers for your new language because they don't understand the domain." It's impossible, moreover, to write a program that is completely correct the first time around, he says. "You always start with something that has errors in it, though they often don't show up until much later in the development cycle. If a bug crops up at that point, when you have 5,000 lines of code, it can be very hard to find it." Consequently, once a program is "code complete"—deemed ready for testing by its developers—software engineers may then have to devote more than half their time to the arduous chore of debugging.

But help is on the way in the form of D2X (pronounced "detox" because it relates to the notion of ridding your program of poisons or defects). D2X is not a program, per se, but is instead classified as a library—a piece of computer code that can be reused by other programs. It is designed to work with existing debuggers (such as GDB or LLDB), serving as a bridge between those tools and a given DSL. A debugger needs information about the program, or programming language, that is to be cleaned up. "Each debugger requires that information in its own particular format, which can be a 400-page document," Amarasinghe says. "If you use D2X, you don't have to worry about that. It's taken care of for you."

With D2X serving as the interface, Brahmakshatriya says, "your program can be debugged using popular debuggers without any modifications to the debuggers themselves." To his mind, that is the main advantage that comes from combining D2X with BuildIt: "If you write a DSL using BuildIt, you don't have to do any extra work. You get a debugger for free, without writing a single extra line of code."

"D2X addresses an inherent contradiction in high-performance software head-on," comments Adrian Sampson, an associate professor of computer science at Cornell University. "On the one hand, domain-specific languages are our only hope for serious improvements in computing efficiency in the modern era. However, making a new debugger for a new language from scratch is hard, and the absence of a debugger is a rational reason that a programmer might reject a 'better' language in favor of a 'worse' one. The great thing about D2X is that it lowers the barrier to constructing a useful debugger for a DSL."

But that's not the end of the story, so far as Brahmakshatriya is concerned. Another feature he'd like to merge with BuildIt, in addition to debugging, is editing, which makes it easier to write a program. Editors, for example, can highlight certain keywords in a document, which can improve its readability. They can perform other functions, such as autocomplete, which automatically fills in text after a small portion is entered.

Brahmakshatriya would like to include profilers along with debuggers and editors as part of the BuildIt platform. "Profilers are like debuggers, but instead of helping you find bugs, they let you assess the performance issues in your program," he says. "If the program is running slower than expected, you can use a profiler to understand which part of the program is bogging things down." Other useful features could be added in the future, he says.

All of these efforts, Amarasinghe maintains, will make the prospect of creating specialized languages much more attractive. "As I see it, there's a huge number of people who support traditional languages—thousands of programmers building tools for C, C++, or Java," he says. "On the other hand, If I am building a simple DSL, I don't have thousands of programmers to provide all that support." But now, with BuildIt and D2X, he adds, "the small guys can get all the things the others get, including debuggers and eventually editors and profilers—the same benefits that come with traditional languages. And you can get that without having teams of engineers writing all kinds of complicated code."

More information: Ajay Brahmakshatriya and Saman Amarasinghe, D2X: An eXtensible conteXtual Debugger for Modern DSLs. groups.csail.mit.edu/commit/pa … 3/ajay-cgo23-d2x.pdf

This story is republished courtesy of MIT News (web.mit.edu/newsoffice/), a popular site that covers news about MIT *research, innovation and teaching.*

Provided by Massachusetts Institute of Technology

Citation: New software tool provides an easier way to debug any domain-specific programming language (2023, April 10) retrieved 26 April 2024 from https://techxplore.com/news/2023-04-software-tool-easier-debug-domain-specific.html