New programming language automatically coordinates interactions between Web page components

December 23 2014, by Larry Hardesty

Credit: AI-generated image (disclaimer)

A Web page today is the result of a number of interacting components—like cascading style sheets, XML code, ad hoc database queries, and JavaScript functions. For all but the most rudimentary sites, keeping track of how these different elements interact, refer to each

other, and pass data back and forth can be a time-consuming chore.

In a paper being presented at the Association for Computing Machinery's Symposium on Principles of Programming Languages, Adam Chlipala, the Douglas Ross Career Development Professor of Software Technology, describes a new programming language, called Ur/Web, that lets developers write Web applications as self-contained programs. The language's compiler—the program that turns high-level instructions into machine-executable code—then automatically generates the corresponding XML code and style-sheet specifications and embeds the JavaScript and database code in the right places.

In addition to making Web applications easier to write, Ur/Web also makes them more secure. "Let's say you want to have a calendar widget on your Web page, and you're going to use a library that provides the calendar widget, and on the same page there's also an advertisement box that's based on code that's provided by the ad network," Chlipala says. "What you don't want is for the ad network to be able to change how the calendar works or the author of the calendar code to be able to interfere with delivering the ads." Ur/Web automatically prohibits that kind of unauthorized access between page elements.

Typing, scoping

Ur/Web's ability to both provide security protection and coordinate disparate Web technologies stems from two properties it shares with most full-blown programming languages, like C++ or Java. One is that it is "strongly typed." That means that any new variable that a programmer defines in Ur/Web is constrained to a particular data type. Similarly, any specification of a new function has to include the type of data the function acts on and the type of data it returns.

In computing the value to return, the function may need to create new

variables. (A function that returned an average of values in a database, for instance, would first need to calculate their sum.) But those variables are inaccessible to the rest of the program. This is the second property, known as "variable scoping," because it limits the scope—the breadth of accessibility—of variables defined within functions.

"You might want to write a library that has inside of it as private state the database table that records usernames and passwords," Chlipala says. "You don't want any other part of your application to be able to just read and overwrite passwords. Most Web frameworks don't support that style. They assume that every part of your program has complete access to the database."

Typing helps with security, too. Many Web development frameworks generate database queries in such a way that someone ostensibly logging into a website can type code into the username field that in fact overwrites data in the database. With Ur/Web, usernames would constitute their own data type, which would be handled much differently than database queries.

Meeting expectations

Typing is also what enables coordination across Web technologies. Suppose that a bit of JavaScript code is supposed to act on data fetched from a database and that the result is supposed to be displayed on a Web page at a location determined by some XML code. If an Ur/Web programmer wrote a database query that extracted data of a type the JavaScript wasn't expecting, or if the JavaScript generated an output of a type that the XML page wasn't expecting, the compiler would register the discrepancy and flag the code as containing an error.

Often, code that isn't explicitly typed still has implicit consistency rules. For instance, if you write a query in the SQL database language that asks

for the average numerical value of a bunch of text fields, the database server will tell you that it can't process your request. To enable Ur/Web to coordinate the flow of data between Web technologies, Chlipala had to create libraries of new data types for SQL, XML, and cascading style sheets (CSS) that embody these rules.

While the Ur/Web compiler does generate XML, JavaScript, and SQL code in its current version, it doesn't produce style sheets automatically. But, Chlipala says, "One thing the compiler can do is analyze your full program and say, 'Here is an exhaustive list of all the CSS classes that might be mentioned, and here is a description of the context in which each class might be used, which tells you what properties might be worth setting.' So, for instance, some particular class might never be used in a position where table properties would have any meaning, so you don't have to bother setting those."

More information: Paper: "Ur/Web: A simple model for programming the Web"

Provided by Massachusetts Institute of Technology