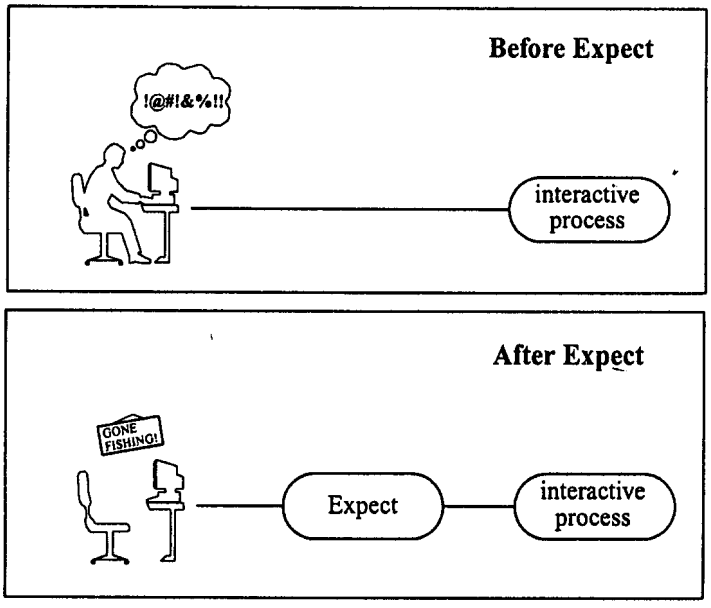
Curing Those Uncontrollable Fits of Interaction

Expect: Curing Those Uncontrollable Fits of Interaction [1] was the seminal paper on Expect, a software tool for automation of other software. The Expect software itself was a solution to a long-standing problem: how to automate software designed to be operated by a human. There are very good reasons for automating software. An obvious one is testing—if a piece of software only works when a human is interacting with it, testing it is very expensive. Another problem is that humans are rather "unreliable." Repeating an interaction for the 10th time, people won't be paying nearly as much attention as they might have on the first and second times.

Many software applications have control languages, preferences, settings, and other mechanisms for providing automated control. Macros are just one modern example of such programmability. However, before Expect each application used a different such language, if any. This meant that users had to learn lots of languages, one for each application. And in many cases, languages were quite limited or totally nonexistent.

Expect combined several ideas. The first was a general purpose control language. No more would people have to learn a new language for every application. The second idea was providing an effective simulation of a human pressing keys at a keyboard. Surprisingly, many programs are quite sensitive to whether a human is there or not—and understandably. For instance, a program intended to accept passwords doesn't want people scripting them since there is a loss in security because embedded passwords are readable by others. Such programs go out of their way to prevent automation. Not surprisingly, many programs could be entirely automated—except for their passwords. But even if the programs were used in secure settings, the programs offered no way of bypassing this manual step.

Perhaps entering a single password doesn't sound time consuming. But imagine creating passwords each semester for fifty thousand students. Or logging in (including entering passwords) to configure several hundred network routers. Or test them. And so on.



Passwords were just one example. Many programs had all sorts of similarly useless requirements for manual operation. Part of the problem arose from software reuse, often in unexpected ways. One of the hallmarks of any long-lived software is its application in all sorts of ways unenvisioned by its authors. A good example of this was Expect's earliest application—in the construction at NBS of a semi-automated factory [2]. This factory was constructed of dozens of one-of-a-kind pieces of hardware and lots of software. Much of the

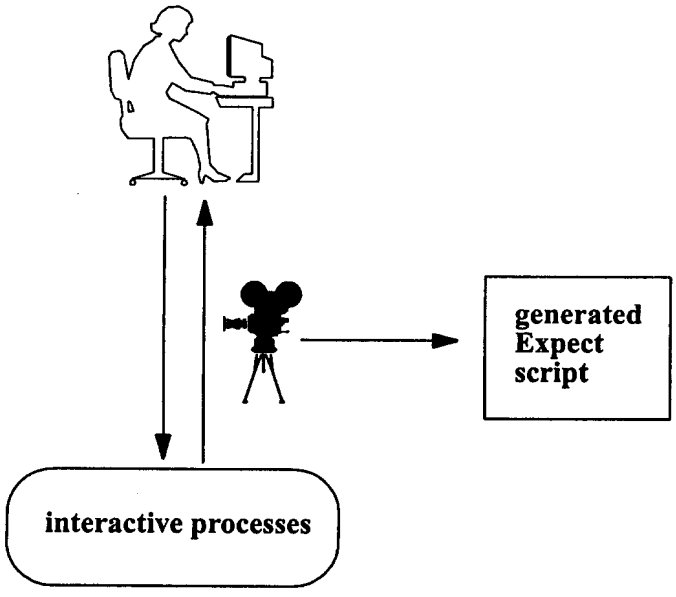
associated software had been written in the context of separate projects, which made perfect sense at the time. It was hard enough to design and develop a robot at all, much less to design and develop a robot that worked in a complex factory. Expect was used as a kind of glue that integrated many of the pieces together. It wasn't a network or a communications protocol, but merely plugged the many assorted holes that the Automated Manufacturing Research Facility software designers had written in for themselves.

At the prestigious 1990 Summer Usenix conference in Anaheim, California, this paper announced the development of the Expect software to a world that was ripe for such automation software. The paper not only described how Expect solved the general problem but it included a dozen classes of software problems that were solved by Expect in order to make sure that readers could walk away with ideas for immediate application to their own environment.

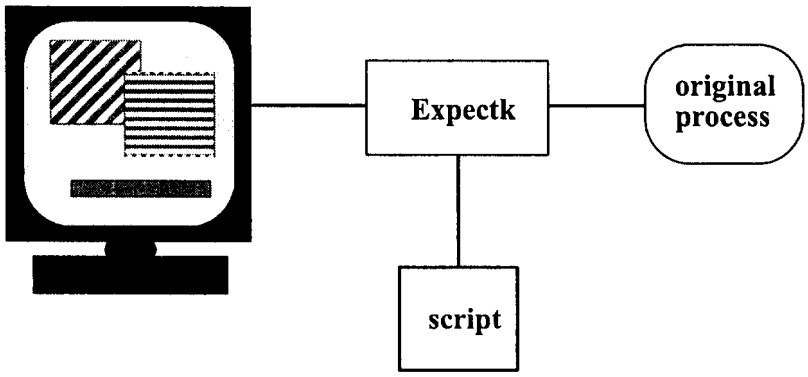
Publication of the paper spurred use of Expect dramatically; NBS saw downloads of Expect hit 4000 in its first year. The bulk of these downloads were .com and .edu sites, but also included 90 U. S. military sites and 170 U. S. federal sites. Two years later, downloads were in the hundreds of thousands of sites, aided by other users making Expect available on network gateways in Australia, Netherlands, Germany, Japan and other countries. Now Expect is bundled with many CD and DVD software distributions and vendor-supplied operating system releases.

Ultimately, the *Curing* paper [1] became a base upon which further exploration and development occurred. Hundreds of other papers, as well as textbooks, graduate theses, technical courses, and other writings followed, each describing Expect as an essential element of various research projects as well as crucial in day-to-day operations. A sampling of Expect's published uses include quality assurance, network measurements, automated file transfers and updates, tape backups, automated queries to multiple heterogenous online databases, computer security sweeps, network router/bridge/repeater/server configurations, test instrument simulation, account administration, stock price retrieval, 500-user simulation, control of unreliable processes, and library management.

Subsequent to the original development of Expect, additional work at NIST continued, including innovative ideas such as automatically generating control systems by watching real interactions (Autoexpect) and wrapping legacy line-oriented systems with modern GUIs without any change to the underlying application (Expectk).

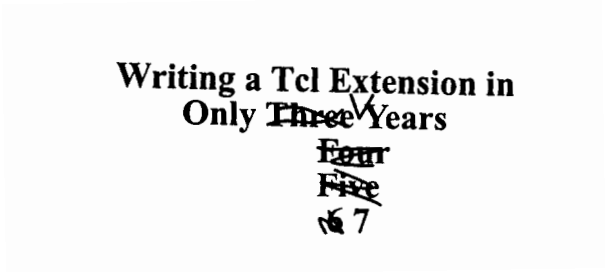


Autoexpect: Automating Automating



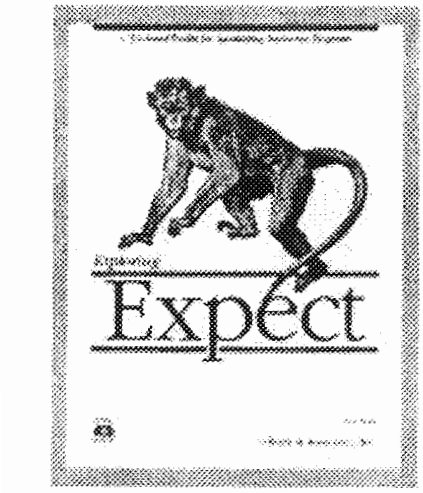
Expectk: Gluing GUIs onto legacy applications

As the figure suggests, the Autoexpect paper had a strange but accurate title. The full title was *How to Avoid Learning Expect—or—Automating Automating Interactive Programs* [3]. As with the Curing title, many of the Expect papers and topics used humor as a very effective writing technique. Perhaps the most amusing title was applied to a retrospective paper on Expect titled:



Because of its odd typography, the *Writing a Tcl Extension* paper was almost impossible to cite correctly, but nonetheless was given the Best Paper Award at the conference where it was presented.

The final development of the *Curing* paper was the publication of the book *Exploring Expect* [4]. Not surprisingly, the book contains significant novel material in its 602 pages. Yet despite being a comprehensive tome, its origins can clearly be traced back to the *Curing* paper. The book appeared in numerous Best Books of the Year lists and has now become the normative reference, superseding the *Curing* paper as the standard citation for the Expect technology.



The paper and book were written by Don Libes, a computer scientist at NBS/NIST. He has written approximately 100 computer science papers and articles plus several textbooks. Besides *Exploring Expect,* he also wrote two classics in the UNIX literature: *Life With UNIX* [5] and *Obfuscated C and Other Mysteries* [6].

Recognition of Expect and the related papers is also demonstrated by various non-government awards that Don earned, including the Award for Excellence in Technology Transfer, Federal Laboratory Consortium (1998), two Best Presentation Awards, USENIX Association (1996, 1997), Best Paper Award, USENIX Association (1997), Federal 100 Award, FOSE & FCW (1993), Innovation Award, International Communications Association (1992), and Tcl Achievement Award, Tcl Consortium (2000).

Prepared by Don Libes.

Bibliography

[1] D. Libes, expect: Curing Those Uncontrollable Fits of Interaction, in *Proceedings of the Summer 1990 USENIX Conference,* Anaheim, CA, June 11-15, 1990, (http://expect.nist.gov/doc/seminal.pdf).

[2] C. Furlani, E. Kent, H. Bloom, and C. Mclean, Automated Manufacturing Research Facility of the National Bureau of Standards, in *Proceedings of the 1983 Summer Computer Simulation Conference,* Vancouver, BC, Canada, July 11-13, 1983.

[3] Don Libes, How to Avoid Learning Expect—or—Automating Automating Interactive Programs, in *Proceedings of the Tenth USENIX System Administration Conference (LISA X),* Chicago, IL, September 30—October 4, 1996, (http://expect.nist.gov/doc/autoexpect.pdf).

[4] Don Libes, *Exploring Expect: A Tcl-Based Toolkit for Automating Interactive Programs,* O'Reilly & Associates, Inc., Sebastopol, CA (1995), (http://www.oreilly.com/catalog/expect).

[5] Don Libes and Sandy Ressler, *Life With UNIX: A Guide for Everyone,* Prentice Hall, Englewood Cliffs, NJ (1989).

[6] Don Libes, *Obfuscated C and Other Mysteries,* John Wiley and Sons, New York (1993).