# PANEL
# What Happened to Integrated Environments?

## Moderator

Hal Hart
TRW
1800 Glenn Curtiss St., Carson, CA 90746 U.S.A.
+1 310 764 6880
Hal.Hart@ACM.ORG

## Panelists

Barry Boehm
University of Southern California
boehm@sunset.usc.edu

S. Tucker Taft
Averstar, Inc.
stt@averstar.com

Tony Wasserman
Software Methods and Tools
tonyw@methods-tools.com

## 1. ABSTRACT

**20 years ago, in 1979, a landmark community-wide process was launched to establish notional requirements for integrated software engineering environments. The resulting "Stoneman" document was published in February 1980. Bred of the software engineering research community and catalyzed by the Government Ada sponsor, this "integrated environment movement" branched out and was embraced widely in the software engineering community in the 1980's as a needed, achievable, centrist approach to accelerate the benefits of disciplined software engineering into mainstream practice. The CASE tool industry bloomed as products integrating lifecycle activities and artifacts emerged, and research evolved to environments integrated by support for emerging, maturing notions of software processes. Yet, at the end of the 1990's, this movement appears to have virtually died, and more and more production software organizations are instead (or again) using old-fashioned stand-alone development tools and struggling to match up tools and their outputs and inputs to do software engineering. The purpose of this panel is to explore the reasons why the integrated environment movement has retreated, whether the recent tools and methods are or are not achieving the goals of integrated environments, whether those goals have been superceded by other goals better served without integrated environments, and what these examinations indicate about future requirements and research needs.**

### 1.1 Keywords

Stoneman, environment, APSE, KAPSE, MAPSE, IPSE, PSE, SDE, SEE

## 2. BACKGROUND

The early "Stoneman" document (aka "Requirements for Ada Programming Support Environments (APSEs),", February 1980) was actually not Ada-specific in its basic concepts. The Stoneman vision included the then somewhat-novel objectives of machine independence of tool suites, lifecycle-wide activity integration (this was before "software process" was a widely accepted notion) including interoperability of separately developed tools, and user interfaces above the old-fashioned command-line styles. As shown in Figure 1, Stoneman articulated an architectural vision of a layered environment with a portability kernel and a fixed set of commonly used tools (a "second layer"), and other project tools (the third layer) would be built on that kernel, optionally utilizing services of the common tools, to achieve those objectives. In addition to virtualizing then-typical operating system services interfaces, a cornerstone Stoneman concept was that the kernel (also called a "framework" or "backplane") go beyond common OS file operations and provide repositories for software development project artifacts and their interrelationships in an entity-relationship-attribute style information base which is organized tailored to the activities, deliverables, and other artifacts of each development project. The goal of this architectural approach was to facilitate making individual tools simpler to design and smaller and easier to fit together at their data-exchange points, instead relying on shared information structures that are provided in common via the project-instantiated framework (not in individual tools) and efficiently shared by all tools in harmony with the project's specified lifecycle activities (process).

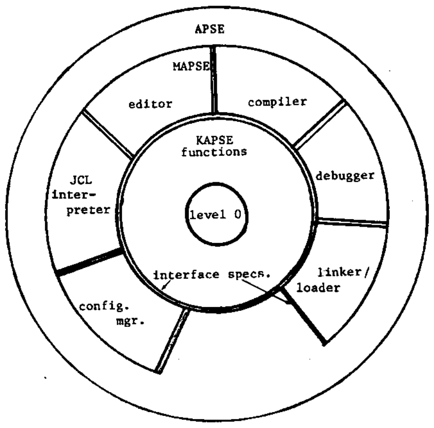

**Figure 1: Stoneman's Layered Architecture Diagram**

The common interpretation was that the Stoneman framework was to be a product separate from individual tools, implemented for every suitable environment-host machine and providing a common set of interfaces for tools to achieve these portability and higher-level interoperability objectives. These frameworks might even be provided by different providers than those building tools for them. For over a decade, throughout the 1980's and into the early 1990's, major efforts occurred in North America and Europe to specify and build Stoneman-style environment frameworks and basic tools suites on them. Most visible of these efforts were the Common APSE Interface Set (CAIS, CAIS-A) and the Portable Common Tools Environment (PCTE), the latter a European-led consortium-style activity including both a framework and basic tools. Interesting related research was occurring in environment-generation technology, the convergence of architectural principles between development environments and applications systems, measurement as applied in environments, reuse-centric environments, and process-driven environments. Environment technology was for over a decade a fundamental, almost centerpiece software engineering discipline, with numerous conferences and workshops on the subject by SIGSOFT and others.

Today CAIS-A, PCTE, other Stoneman environments and many of the related efforts, after many thousands of person-years of research and development, are relics of history or at least not being realized in the state-of-the-practice, for reasons that the panel will briefly explore. Some of the Stoneman objectives have been at least partially fulfilled by different approaches than envisioned 20 years ago (e.g., external data transport standards and commercial databases used as the data repository in tools, common object libraries, ORBs, etc.), and some remain illusive and vexing today (e.g., total portability of tools and difficulty in effectively integrating tools from different vendors). New imperatives for development environments have arisen -- internet- and web-based interactions, distributed collaboration, agent technology, visualization, new security issues, accommodating reuse- and COTS-based developments where integration is the key developer value, broadening lifecycle perspectives to include system engineering and maintenance, etc.

The panel will compare the objectives of twenty years ago to today's needs, and cast opinions about in what ways today's tools and environment offerings are better or worse than Stoneman-style and process-driven environments would have been on today's machines. And, panelists will be asked to conjecture on what visions, objectives, and approaches they would write different or the same in a "Stoneman-2000" -- in other words, chart tomorrow's research issues in software development support technology.

## 3. ABOUT THE PANELISTS

Barry Boehm is Professor of Computer Science and Director of the Center for Software Engineering at USC. Among his activities at TRW in the 1980's were sponsorship of two widely cited internal environment developments, "SPP" and "Quantum Leap." (See Barry Boehm's keynoter description for more biographical data.)

S. Tucker Taft is Technical Director of the Distributed Information Technology Division at AverStar, and Chief Architect for AverStar's Ada 95 technology. At Harvard in 1970, he managed the first Unix system outside AT&T. At Intermetrics in 1980 he participated in the development of the Ada Integrated Environment (AIE) for the Air Force. More recently, he led the Ada 9X language design team, culminating in the1995 approval of Ada 95 as the first ISO standardized O-O programming language.

Anthony (Tony) Wasserman currently is president of his own consulting business, Software Methods and Tools. He is a pioneer in the software engineering and CASE communities as a co-founder, first elected Chair in 1976, and long-running officer of ACM SIGSOFT; and an early tool builder. In the 1970's Tony helped create the vision of "integrated development environments" -- suites of tools providing automated assistance for multiple successive phases of activities in what would later be called "lifecycle processes." Tony founded his first company (IDE), and its major product, *"Software Through Pictures" (STP)*, set the standard for CASE tools in the 1980's.

## 4. REFERENCES

[1] Department of Defense Requirements for Ada Programming Support Environments, "Stoneman", February 1980. Edited by Professor John N. Buxton.