PERSONAL COMMENTARY

William Bricken
April 2005

My resume is unusual. I have intentionally sought to develop a diversity of skills, under the assumption that experiences from many perspectives would eventually integrate into something resembling wisdom. Some experiences, such as carpentry (three years) and executive business leadership (three years), have not suited me well. Others, such as graduate classroom teaching (six years) and advanced programming and computational logic (twelve years), have been delightful and engendered deep expertise. Currently I can demonstrate wide cross-disciplinary knowledge, an exceptional ability to integrate and synthesize, a keen intuition about the future, mastery of both programming and teaching, personal contentment, and a love of life.

Each of the phases of my career is marked with the completion of a significant product. I believe strongly that knowledge and activity are tightly coupled, and have endeavored to concretize my own learning within a context of constructive production. Thus, the relatively sparse publications which mark my abstract work are each seminal and comprehensive (egs: The VEOS Project, Inclusive Symbolic Environments, and Boundary Logic). They are also relatively obscure, since I have been motivated by personal learning and teaching rather than by building either an academic reputation or a public portfolio. Much of my technical work within business environments remains trade secret due to potential commercial value. My work building research environments and technology prototypes has achieved national prominence.

Leadership

I have always found myself to be in a leadership position, progressing through corporate principal scientist (Advanced Decision Systems), corporate lab director (Autodesk), university lab principal scientist (Human Interface Technology Lab), and start-up company chief scientist (Bricken Technologies Corporation). As an administrator, I am invariably clear and honest, but occasionally lacking in the nuances of diplomacy. My analysis of situations is deep, most often accurate, and somewhat slow to develop. Almost always I take on large, complex tasks, enjoying the invigoration of difficult intellectual challenges.

To gain respect as a technical leader, it is necessary to know a particular technical field at least as deeply as one's colleagues. When I began programming as a professional, I achieved local recognition among the programming community associated with Stanford University. My work included a database interface that learned the user's intentions, a deductive engine based on an entirely new model of logic, and a parallel logic optimization architecture. When I established the Autodesk Research Lab, within a year we produced and demonstrated the first publicly accessible virtual reality system, precipitating a wave of excitement that altered the general public's

conception of computer graphics. Technical respect has permitted me to attract some of our country's best programmers to the labs I have founded.

My management style is strongly consensual, participatory, and very tolerant of diversity. I am rarely frustrated. I believe that the job of management is to remove administrative burdens from co-workers, in order to make their tasks easier and more enjoyable. I am skilled in leading groups toward consensus, and expect people to work hard toward common goals that they develop through mutual interest, cooperation and negotiation.

Vision

Over the years, I've developed a remarkable vision for seeing the whole technical picture, which has lead to trust and respect in the construction of both business alliances and educational curricula. The most important consequence of vision is to be able to scope concrete, achievable tasks, to be able to recognize what is easy, what is hard, and what is impossible. This is particularly valuable for management of funding and resources, and in the estimation of task complexity and resource allocation. I have learned how to select appropriate topics for funding applications, how to accurately assess costs and expected dividends, and how to judge and control the level of effort to meet deliverables and deadlines.

Vision requires both breadth and depth of experience. I can see connections across disciplines in computer science primarily because, over the last 13 years, I have designed and/or developed significant successful systems in each of these areas:

```
-- circuit synthesis
      logic optimization and reconfigurable computing (at IRC and BTC)
-- operating systems
      Cyberspace virtual environment (Autodesk)
      Virtual Environment Operating Shell (HITL)
-- programming languages
      Losp, an algebraic logic language (ADS)
      ConMan, a constraint management language (ADS and Boeing)
-- parallel architectures
      parallel deduction engine (ADS)
-- computer graphics
      3D and fractal modeling systems (Autodesk)
-- agent architectures
      entity modeling (HITL)
      distinction networks (IRC)
-- computer-based tutoring systems
      algebra tutor (Stanford)
      EduSpace (Oz)
-- human-computer interface
      intelligent interface (ADS)
      virtual environments (Autodesk and HITL)
```

I can see connections across academic disciplines because I have achieved practical expertise in curriculum design, educational psychology, statistics, research methodology, and computer science. And I can integrate this abstract knowledge into the management of practical experiences because I have developed much of it through the construction of grounded, practical systems (business research labs, entrepreneurial companies, logical optimization and virtual reality systems).

Yet vision requires more, it requires learning from the great minds of our time. In this I have been blessed with the opportunity to work with and learn from truly excellent software theoreticians and developers: John McCarthy, Mike Genesereth, Doug Lenat, and Bill Clancey at Stanford; Alan Kay and Brenda Laurel at Atari; John Walker, Ted Nelson and Stephen Wolfram while at Autodesk; Dick Shoup and Andrew Singer at Interval Research Corporation.

Entrepreneurship

I have served as Chief Scientist for two start-up companies. The first had a business plan to develop virtual reality systems for health and exercise applications. My responsibility was to develop a technical capability, and my team succeeded in doing that. The price point for delivering commercial systems was too high for the marketing group to succeed, and after raising $600,000 the company closed. The second start-up was formed to commercialize my work in boundary mathematics, as applied to semiconductor design and optimization. It provided me with the opportunity to spend three years intensively programming. I loved the opportunity to develop novel, high-performance algorithms for exceptionally difficult optimization problems. The company was formed, unfortunately, just after the dot-com bubble burst, so attracting funding proved to be very difficult. We finally closed the corporate doors early in 2005. Building software in start-up companies is arduous, due to limited time resources. Errors are too costly to be acceptable, and I thrived in these difficult circumstances. I deeply enjoy the challenges of high performance programming because I enjoy designing architectures and algorithms that require extremely high quality. I've developed a suite of unique skills that apply widely, and I like to use them directly to solve business problems.