



The Menlo Institute LLC

Heroic Efforts

Heroics Should Describe Soldiers not Software Engineers

**Thomas Meloche © 2002
Fellow, The Menlo Institute**

**The Menlo Institute LLC
212 N. Fourth Ave
Ann Arbor, MI 48104
www.menloinstitute.com
(734) 665-1847**

Successful Delivery

It is all too common to encounter organizations that deliver software by working the development team 50, 60 or 70 hours per week prior to a release. The most honored programmers, the heroes, are the ones who work the longest hours and deliver the largest chunks of software. Unfortunately, the systems delivered after such an initiative are often buggy, prone to crashing, difficult to maintain and support. Is this successful delivery? Were the programmers really heroes?

The Java Factory

One bright summer day while visiting a favorite client, I found myself pulled into the middle of an impromptu sales meeting. My client, who was working hard to improve their software development process, was telling their prospective customer about the new Extreme Programming process and how it was delivering such successful results.

Unfortunately, my client's customer did not seem all that impressed. Perhaps the Extreme Programming process was scaring them, or perhaps it was the strange look of the giant collaborative development space we were meeting in; an abandoned factory affectionately nicknamed "The Java Factory."

Trying to help, I noted that we were making the software development process more of an engineering discipline. I told them we wanted predictability and repeatability in producing quality software and did not want to rely on heroic efforts.

They replied that *all* of their software projects were delivered as the result of heroic efforts. They proudly proclaimed that their engineers often had to work long hours all of the way up to the last minute, attempting to release a system. They seemed to like that their process demanded heroic efforts. They were bragging about it.

I was appalled.

Now don't get me wrong, I am all for heroic efforts. Firemen may be heroic, policemen may be heroic, soldiers and lifeguards may be heroic. Their jobs occasionally demand it. But none of these people want their jobs to be heroic. Heroics imply risk, danger and a high likelihood of failure and disaster. That is why the actions are heroic. I don't know about you, but I prefer not to build software with risk, danger and a high likelihood of failure and disaster.

No Heroics Allowed

I don't want my software engineers to be heroic. Heroics are not allowed on our functioning teams. We don't want programmers working long hours developing software in clear violation of best practices. It is not a good way to run a delivery organization. It is a good way to fail.

Is heroic software user friendly?

Is heroic software properly tested?

Is heroic software going to work? How often? For how long?

If I am flying in a plane, I don't want to hear that the navigation software was built with heroic efforts. If I am buying a car, I don't want to hear that the anti-lock breaking software was built with heroic efforts.

Heroics should describe soldiers, not software engineers.

Intelligent, disciplined, careful and controlled are words that should be used to describe our software development efforts, along with inventive and creative, but not heroic.

Managers who describe their software projects as heroic should be ashamed. The entire Menlo Institute is dedicated to eradicating the idea that software development should be heroic.

Hard working, yes.

Successful, yes.

Heroic, no.

Postscript

The manager who proudly described his software process as heroic produced payroll software, so you better check your pay stubs. Does heroic payroll software calculate your pay accurately?

About the Menlo Institute
 Founded in 2001 by Thomas Meloche, Richard Sheridan, James Goebel and Robert Simms, The Menlo Institute specializes in Software Development Process and Methodology with a special focus on an agile software development environment called The Software Factory.

The Software Factory
 The Software Factory is a complete agile software development methodology created at Menlo. The Software Factory combines best practices from Alan Cooper's Interaction Design, Kent Beck's Extreme Programming, The Rational Unified Process and Six Sigma. The Menlo Institute has unique insight from real-world experience on how these practices can work together effectively.



**Thomas Meloche
 President
 The Menlo Institute**

Thomas Meloche is President of The Menlo Institute LLC and a founding member of Menlo Associates LLC. His passion is software development best practices and building successful software teams. Before founding The Menlo Institute, Thomas was the Director of Engineering at AppNet, Inc. For over seven years, he led a consulting staff that grew from

30 people to over 240 people which generated revenue of over 100 million dollars. As a result, he is intimately familiar with the real problems that befall real software development projects. He also knows how to rectify them. During his tenure his delivery organization had a remarkable 100% successful delivery record.

Thomas has a B.C.E. in Computer Engineering from the University of Michigan, and especially enjoys providing training back to his alma mater. He has taught classes and provided mentoring for both the University of Michigan Law and Medical Schools.

Menlo Institute Classes by Category		
Extreme Programming (XP)	Foundation Courses	Rational Unified Process (RUP)
Introduction to XP Coaching XP Teams Writing Story Cards XP BOOT Camp / Immersion	Secrets of Software Success Six Sigma Software Building a Software Factory Object Technology Overview	Managing with RUP-Lite Capturing Reqs With Use Cases Practical UML RUP BOOT Camp / Immersion
Requirements Gathering	Object Technology	Project Management
Capturing Reqs With Use Cases Writing Story Cards	Object Technology Overview Obj. Oriented Analysis & Design Advanced Obj. Oriented Design	Managing with RUP-Lite Software Project Mgmt Six Sigma Software