The Victim Trap

The TSP Symposium

September 23, 2008

Watts S. Humphrey and
The Software Engineering Institute
Carnegie Mellon University

A War Story - 1

The team had 9 software and 4 hardware engineers.

They were charged with building a new hardware-software system.

This was to be a new communications line testing device.

Management told the team the product was critically needed in 9 months.

A War Story - 2

Developing the prior product had taken two years.

Would this be another crisis project?

- late
- over cost
- months of testing
- lots of defects

Or could the team somehow succeed?

The key question: Will this team fall into the victim trap?

Carnegie Mellon
Software Engineering Institute

Agenda

Being a Victim

The Nature of Software Work

Breaking from the Victim Trap
- Attitude
- Skill
- Courage
- Credibility

Conclusions

Being a Victim

Ask developers why their projects are troubled, and you will typically get several reasons.

- Changing requirements
- Overly aggressive schedules
- Management pressure
- Inadequate resources

Victim Talk

These answers are different ways of saying

- It wasn’t my fault.
- They did it to me.
- Isn’t life awful?

This is victim talk.

The Causes of Victimhood

Victims feel powerless to change the situation.

Software developers typically feel that

- they work to dictated schedules
- their processes are imposed
- somebody else controls the requirements
- their assignments can change without warning

What Makes Victims

People act like victims for two reasons.

- They are powerless.
- Someone has actual or perceived power over them.

Software people feel like victims because of the nature of their work.

This is the way software work has always been managed.

The Nature of Software Work

Software work is challenging.

- Every project has changing requirements.
- Schedules are always aggressive.
- Management's job is to exert pressure.
- Resources are always tight.

This is the way the software world has always been.

And it is the way it always will be – unless we change it.

Carnegie Mellon
Software Engineering Institute

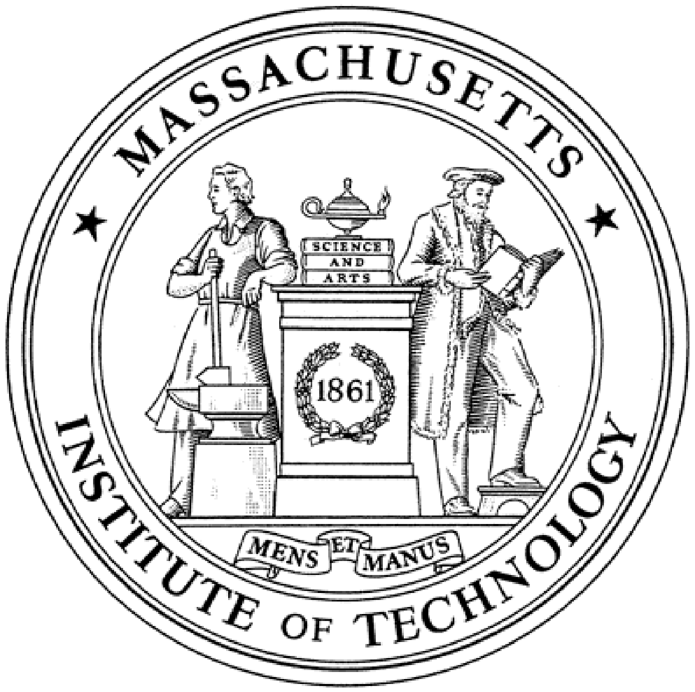
Successful Engineers

But engineers in other fields are not victims.

How come?

Truly successful engineers are optimists.

- They are convinced the job can be done.
- They can think of thousands of alternate things to try.
- They never give up.



The Nature of Engineering

Other developers build new and advanced products.

Their work also involves

- exploring the unknown
- overcoming obstacles
- being creative and resourceful

But they also do quality work for predictable costs and on committed schedules.

Software Differs from Hardware

Software and hardware work both involve design.

The big difference is implementation.

Manufacturing represents most of the hardware costs.

For software, manufacturing costs are generally small.

The manufacturing discipline shows hardware engineers how to avoid the victim trap.

Breaking from the Victim Trap

The way to break out of the victim trap is to assert yourself.

While this may sound easy, it is not.

There are four things you must have.

- The right attitude
- The proper skills
- Courage
- Credibility

Only then can you assert yourself.

And if you don’t assert yourself, you will remain a victim.

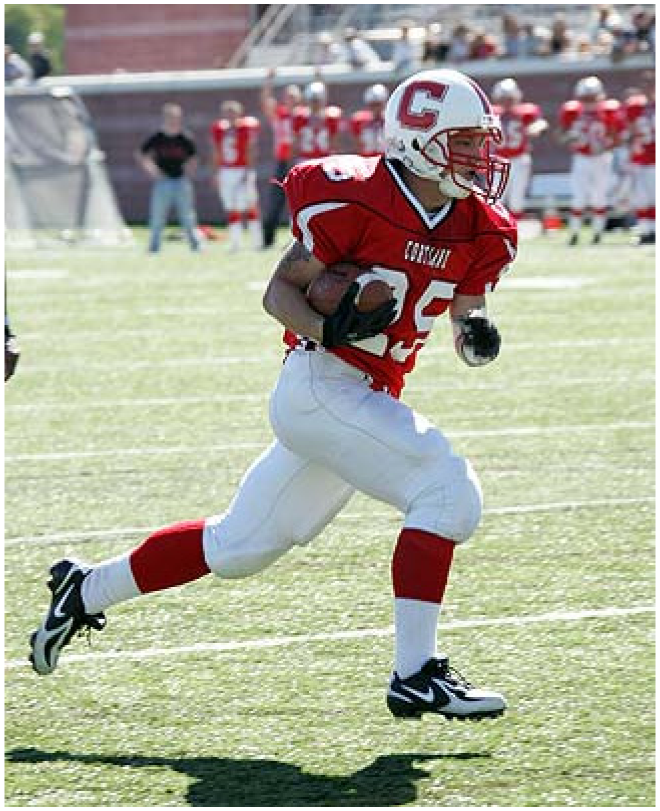
Carnegie Mellon
Software Engineering Institute

The Importance of Attitude

Winners generally win.

Victims feel like losers.

Losers usually lose.

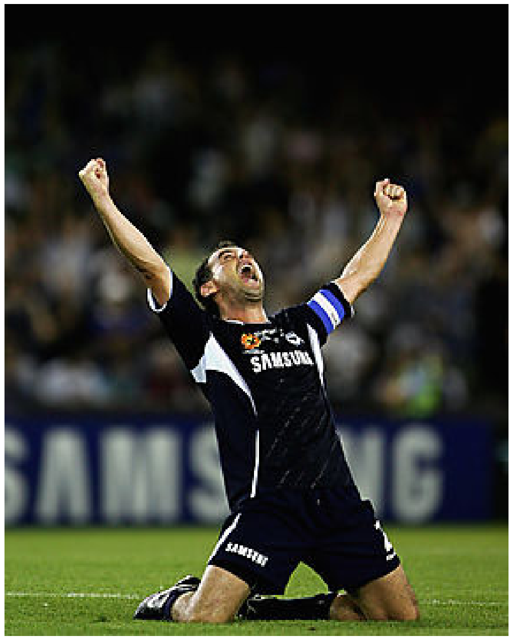


The Desired Attitude

For developers to consistently win, they must act like winners.

To act like winners, they must

- be confident that they can win
- believe that success is up to them
- feel in control of the outcome
- feel personally responsible for the outcome



Changing Attitudes

To act like a winner, you must feel like a winner.

But to feel like a winner, you must be a winner.

Software teams usually lose.

Cheerleading alone will not make winners.

Being a Winner

To be a winner, you must meet your commitments.

To meet your commitments, you must

- resist unrealistic commitments
- negotiate realistic commitments

This takes skill, courage, and credibility.

The Need for Skill

To resist unrealistic commitments, developers must

- recognize unrealistic commitments
- be able to make realistic plans

To recognize that a commitment is unrealistic, developers must

- understand that all unplanned commitments are unrealistic
- insist on making a plan before agreeing to any commitment

To do this, they must

- know how to make estimates and plans
- have the data to make accurate estimates

What Next?

We must have a winner's attitude.

We must also have planning skills.

But it takes courage to negotiate with management.

Building Courage

There are three parts to building courage.

First, you must know what you are talking about.

Second, you must have self confidence.

Third, you need the support of allies.

The PSP training and TSP launch provide all three.

A TSP Industrial Project

The project was to develop communications test equipment.

	Plan	Actual
Size - KLOC	110	89.9
Effort - hours	16,000	14,711
Schedule - weeks	77	71
Defects per KLOC		
Integration and system test	1.1	0.6
Field trial	0.0	0.02
Customer use	0.0	0.0

Building Team Capability

Everyone on the team must have skills, knowledge, and self-confidence.

Why everyone?

Suppose only one person on the team had the needed skills and knowledge.

- No one else would have self confidence.
- The team’s ability to negotiate a realistic commitment would depend on one person.

That is the problem with most software teams.

Carnegie Mellon
Software Engineering Institute

Needed Knowledge and Skills

Basic knowledge

- What plans are
- Why plans are important
- The importance of data in planning

The skills

- How to gather data
- How to estimate with data
- How to build plans from estimates

Self-Confidence

The second critical step is building self-confidence.

To build their self-confidence, teams must

- know what management wants
- know how to meet that need
- have a plan for the work
- have confidence in the plan

The team members must also have trust.

- In the entire team
- In management

Getting Allies

The third critical need is for allies.

Allies are important for two reasons.

- Safety in numbers
- Team commitment

TSP teambuilding provides allies.

- A sense of belonging
- The support of coworkers
- A disciplined environment
- A common commitment to team goals

Then it is not just the team leader’s plan, it is the team’s.

TSP Teambuilding

To build teams, groups must work under pressure to accomplish risky and challenging tasks.

The TSP launch task is to build the team’s plan.

The risk is not meeting management’s needs.

Everyone has the knowledge and skill to do the work.

The result is a cohesive group and a detailed and accurate plan.

The Final Step

We have discussed

- the winners attitude
- planning skills
- the courage to resist unrealistic commitments

What about credibility?

Carnegie Mellon
Software Engineering Institute

Credibility

TSP teams get credibility by

- making detailed plans
- standing behind these plans
- committing to meet their goals
- being cohesive and committed

Management is impressed by the team's plan.

But the team's attitude and self-confidence are most convincing.

Carnegie Mellon
Software Engineering Institute

Conclusions

The TSP enables developers and their teams to break out of the victim trap.

Initially, TSP teams have credibility because they have produced detailed plans.

To retain this credibility, these teams must consistently meet their commitments.

For More Information

Visit the PSP or TSP web sites
- http://www.sei.cmu.edu/psp/
- http://www.sei.cmu.edu/tsp/

Contact a PSP transition partner
http://www.sei.cmu.edu/collaborating/partners/trans.part.psp.html

Contact SEI customer relations
Software Engineering Institute, Carnegie Mellon University
Pittsburgh, PA 15213-3890
Phone, voice mail, and on-demand FAX: 412/268-5800
E-mail: customer-relations@sei.cmu.edu

See the book
Winning With Software: an Executive Strategy, by Watts Humphrey, Addison-Wesley, 2002