

First Break All The Rules

The True Story of a
Death March Project

Mary Poppendieck
www.poppendieck.com
www.leanprogramming.com

The Players

State Agency

Needed two legacy registration systems to be replaced by July 1

Prime

Under contract to modify systems developed for other states

Sub

Provided analysts and programmers

PM

Provided project management

The Problems

Time

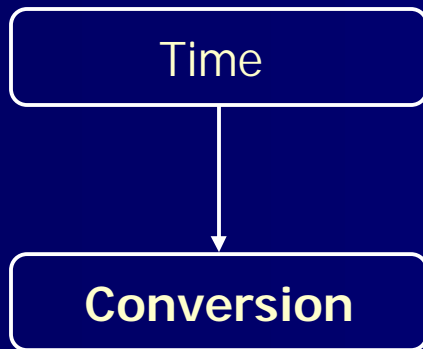
Half the timeline had produced three inches of approved requirements.

The state being used as a base system was changed, rendering the requirements obsolete.

New business rules were mandated by law for Registration System 1, to go into effect July 1.

Registration System 2 was running on an expensive computer whose lease expired July 1.

The Problems



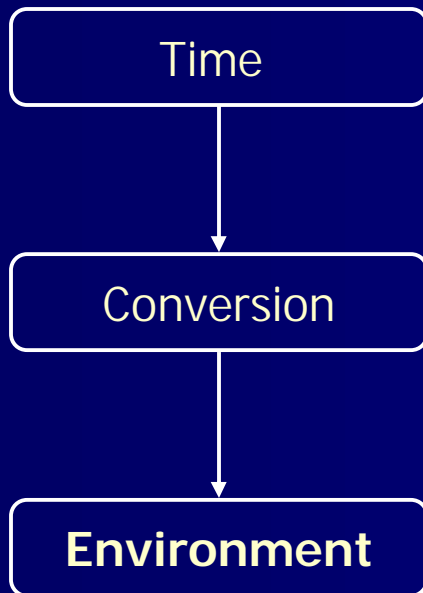
The new base system was incomplete, undocumented, and consumed the prime's resources.

There was no working system to provide a basis for a gap analysis.

Programmers had to reverse engineer complex, fragile VB code.

Converting the base system from Oracle to SQL introduced devious database errors.

The Problems



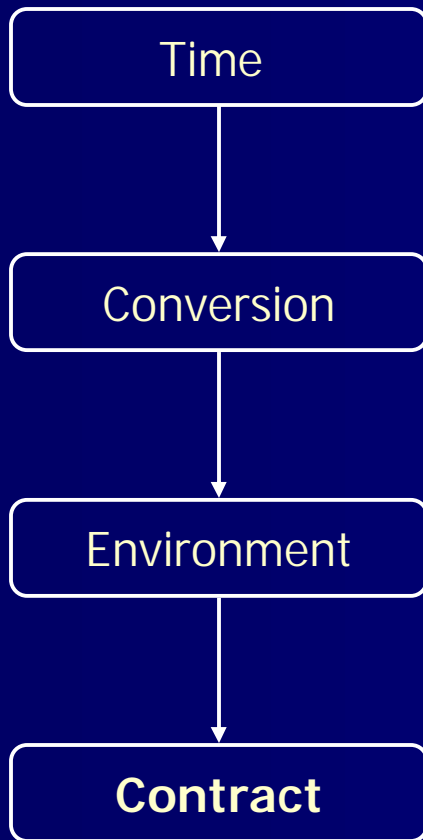
Junior business analysts were expected to provide detailed, written instructions for junior developers in another country.

There was little testing capability in place; un-tested code was delivered in order to meet the aggressive schedule.

Complex Thin Client environment was not replicated at the programming site.

A third party imaging system (different from base system) required extensive integration at the customer site.

The Problems



The contract was fixed price, fixed scope, fixed deadline. There was no way the contract could be met.

The contract called for a traditional waterfall approach, with payments tied to documentation delivery.

It took four months to get sign-off on requirements, which were then put under strict change control.

A lawsuit was a strong possibility.

Attacking The Problems

Honesty

This is not going to happen.

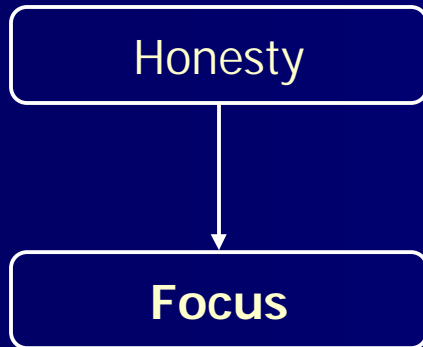
What may be possible is meeting the legal deadline with System 1.

Reaction:

- State Project Manager
 - Very Positive
- State CIO, Sub CEO
 - Very Negative

It took two months and a good idea for the CIO and CEO to accept reality.

Attacking The Problems



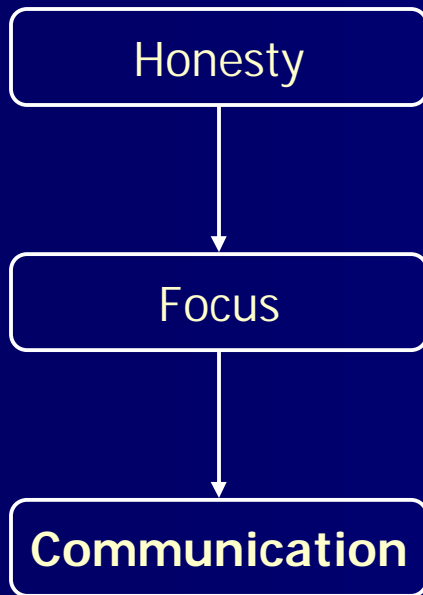
Do nothing except produce code.

Change control abandoned in favor of a one sentence scope control agreement.

No time was wasted on finger-pointing. We agreed that in any situation, there was blame on each side, so we would work together to resolve any problems.

The schedule was immediately changed to incremental deliveries of features, and all resources were focused on the next scheduled delivery of code.

Attacking The Problems



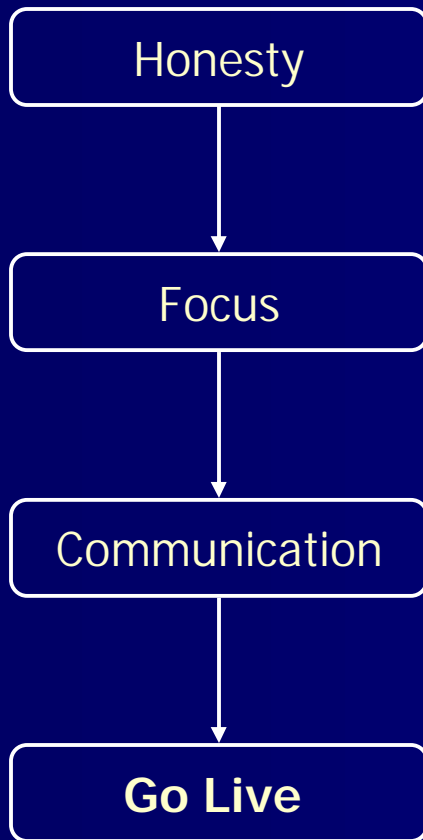
Get people talking face-to-face.

The analysts were sent overseas to spend a month with the developers.

Testing, integration with imaging and environmental problems had to be dealt with at the customer site, so developers were brought there from overseas.

A great effort was made to get the prime's developers to go to the sub's development site or to the customer site. This effort largely failed.

Attacking The Problems



Now comes the risky part.

The delivered code was untested, and as we later found out, it did not work.

The legacy conversion was untested, but in fact it had only a few problems.

The CIO and CEO wanted to go live at all costs. The technical people did not think this was a good idea.

The old system 'went dead' on June 29. The new system 'went live' on July 23.

You've Got To Be Kidding

- The system was less than 50% operational. We could add registrations, but not search the database. The fiscal system was not done. The legacy conversion was lightly tested.
- Errors in filling orders to search the database had legal ramifications. Delays had legal implications.
- All future work would have to be done in the context of a production environment. The database and code integrity needed protection. Pressure from production users could dominate decisions.

How To Develop In A Production Environment

Pause

No More Death March.

Stop working nights and weekends.
The deadline is PAST.

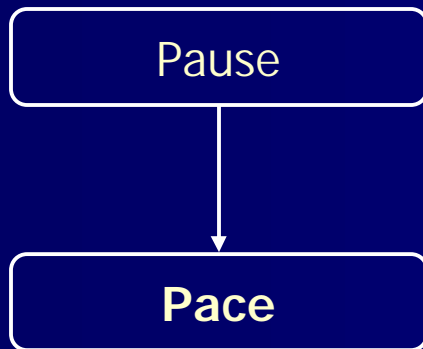
Put together a complete Punch List
(Similar to the SCRUM Backlog).

Have the customer prioritize the list.

Practice one minute scope control.

Work down the list until it's done.

How To Develop In A Production Environment



Weekly Production Releases.

Friday: Development team meets with customers; identifies items at the top of Punch List they can commit to finishing.

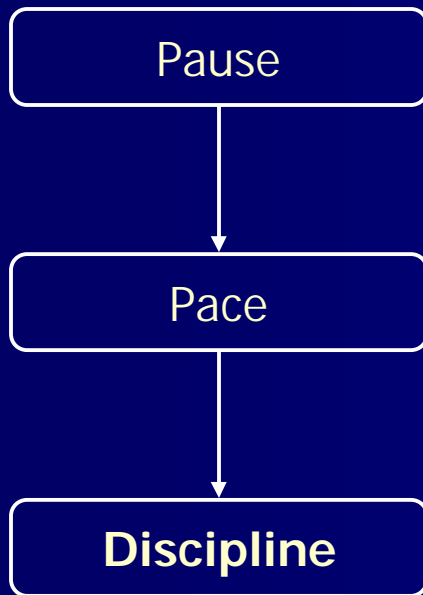
Wednesday night: 1st production build.
Thursday morning: 1st regression test.

Thursday night: 2nd production build.
Friday morning: 2nd regression test.

Weekend: Repeat if necessary.

Monday: Release build to production.

How To Develop In A Production Environment



We're in production now.

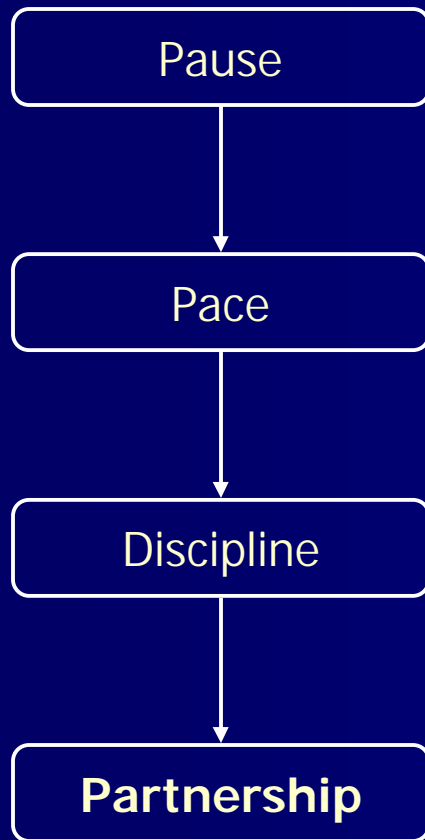
Test scripts must be complete and updated with each new release.

Dual environments (production / test) must be rigorously maintained. The development team must not have access to the production environment.

A top notch DBA is essential.

Maintain the ability to back up.

How To Develop In A Production Environment



It only works if the customer is happy.

Check with the users every day to see how things are going. Keep them informed on how development is going.

Follow up on repeatable problems; they usually indicate defects in the code.

Do not waste too much time on single instances; they are usually user errors.

Insulate developers from production issues unless system integrity is at risk.

A Customer From Heaven

- Department managers were deeply involved.
- Testing was done on-site and regression testing was done by users.
- Users put up with innumerable workarounds when features were missing.
- The customer project manager consolidated the various user priorities into a single prioritized list.
- The department managers kept their end of the agreement on one-minute scope control.
- We never did find the time to assign blame; we worked together to resolve each problem.