Demands That Stress Patterns

Tobias Koch and Sven Eisenhauer

Fachbereich Informatik
Hochschule Darmstadt

Seminar Systems Thinking, WS2009/2010

Content

Content

Square law of computation

Definition

"Unless some simplification can be made, the amount of computation to solve a set of equations increases at least as fast as the square of the number of equations."

Corollary

The "computer" needed to control a system has to become 4 times more mighty as the system size doubles.

Square law of computation

Definition

"Unless some simplification can be made, the amount of computation to solve a set of equations increases at least as fast as the square of the number of equations."

Corollary

The "computer" needed to control a system has to become 4 times more mighty as the system size doubles.

Example

- Even trained chicken can play perfect tic-tac-toe.
- Nobody can play perfect chess.
- Although chess is a perfect game with all information given and a fixed board size it is too big for every existing computer, human or artifical.

What makes management difficult

- Management is similar to controlling a game: Get from A to B.
- Management is much more complex: Not a perfect game.
- Not all information is available.
- Board size is not "fixed".

What makes management difficult

- Management is similar to controlling a game: Get from A to B.
- Management is much more complex: Not a perfect game.
- Not all information is available.
- Board size is not "fixed".

What makes management difficult

- Management is similar to controlling a game: Get from A to B.
- Management is much more complex: Not a perfect game.
- Not all information is available.
- Board size is not "fixed".

What makes management difficult

- Management is similar to controlling a game: Get from A to B.
- Management is much more complex: Not a perfect game.
- Not all information is available.
- Board size is not "fixed".

Content

Size/Complexity Dynamic

Fact

- *Human brain capacity is limited.*
- *Complexity of a program grows by the square of its size.*
- *Augmented ambitions after success lead to more complex products.*

Corollary

- *Sooner or later a program will become too big to be handled by a human brain.*
- *Development of more complex products need to be "simplified" by methological software engineering.*

Other forms of this dynamic

- Fault/Location Dynamic
- People/Interaction Dynamic

Fundamental Problem

Fact

Ambitious requirements can easily outstrip even the brightest developer's mental capacity.

The Right Tool for the Job

- To battle the Natural Dynamics of large projects, the right methods of human intervention have to be applied.
- The right choice of technologies, working styles, methodologies, ... may depend on the nature and complexity of the problem at hand.

Compositing Engineering Methods

- Because different methods may excel at different problems or problem sizes some organizations try to combine methods.
- Pattern 3 (Steering) managers will readily use a toolkit of engineering methods.
- Managers in a "blaming environment" will rather stick to one "standard" way of doing things, so they cannot be blamed for making the wrong choice.

Taking Risk into Consideration

- Different methods will have different risk levels, meaning the probability of success will differ depending on problem size.
- Important: The risk rate doesn't say anything about the cost.
- Human beings learn: the success rate of a method will increase, when applied multiple times.

The Threat of Change

- Managers may choose not to implement new methodologies in order to not jeopardize their careers.
- Solutions:
  → Move decisions to a higher management level.
  → Run a pilot project at minimal size.
  → Try to reduce the criticality of the very first project.

Helpful Interactions

1. Tackle variability by bringing all the dynamics that are part of the engineering process under control one by one (using Intervention Dynamics).

The Helpful Model

2. Realize that different people develop *different models* to measure and control a situation, leading to potentially very different results.
   → "No matter how it looks, everybody is trying to be helpful."

More Helpful Interactions

1. It's hard to erase existing ineffective behavioral patterns.
   → Try not to erase them, but instead add new patterns that are more effective, ultimately overlaying the original patterns.
2. Variation: It's hard to change someone's perception of reality.
   → Propagate adoption of new models of thinking to open people's eyes.

Outside Influence

Fact

Outside influence contributes to the instability of a software development process

More customers increase development load

- Greater number of requirements ==> more conflicting requirements
- Greater system complexity
- More labor to deal with conlicting requirements
- Labor to deal with customers

Corollary

Nonlinear Size/Complexity Dynamic that can lead to the collpase of a cultural pattern

Two-Way-Relationship

Between Software Organization and Customer

From Software Organization To Customer

- Software

From Customer To Software Organization

- Resources
- Requirements

Corollary

A controller is needed to control flow of requirements, resources, outputs and randomness to Software Organization

User != Customer

Difference between User and Customer

- User is everyone affected by the system
- Customer defines quality

Effective Customers

- Interact with Software Development
- Marketing function

Marketing

Functions of Marketing

- reduces number of effective customers by standing between Software Development and customers.
- It filters inputs and outputs.

Can be dangerous

- Near the core of the Software Development system.
- Uncontroled input

Interruptions of work

How much cost interruptions?

- An increased number of customer also increases the number of interruptions of work.
- E-factor=Uninterrupted Hours / Body-present Hours
- Total time = Interruption time + reimmersion time (phone call: 5+15 = 20)

Meetings are even worse

- more customers mean more meetings with more people
- more people increase number of interruptions
- number of interruptions increase wasted time (depends on avg. length of interruption)

Hardware configurations

- More customers lead to more hardware configurations in production.
- Number grows exponentially, so not every possible configuration can be tested, tests become more complex.
- This leads to less coverage and more faults
- Result in more time needed for fault repairing.

Releases

- More customer result in more releases and versions used by customers
- Leads to more labor in maintaining the hole software product.
- More reported faults balance the management tenendcy to increase releases cycles. So two releases per year are common to many software organizations.

Conclusion

Essence

- Customers influence the demand of a certain cultural pattern in a software organization.
- Customers influence the size Software projects.
- Management cannot handle projects beyond a specific size perfectly
- Reducing number of effective customers is a common strategy of software organizations to reduce disturbances on the software organization.

Thank you...

... for your attention!

Conclusion

Essence

- Customers influence the demand of a certain cultural pattern in a software organization.
- Customers influence the size Software projects.
- Management cannot handle projects beyond a specific size perfectly
- Reducing number of effective customers is a common strategy of software organizations to reduce disturbances on the software organization.

Thank you...

... for your attention!