

Project Management Course Book

T++ Technical Skills Training Program

CUNY Institute for Software Design & Development (CISDD)

New York Software Industry Association (NYSIA)

Aron Trauring

Class 6 October 21st, 2004

Class Agenda

1:30-3:00 Agile Management Theory - Part II (Presentation)

3:00-3:15 Break

3:15-4:00 Exercise 6 — Agile Management Theory in Practice

4:00-4:15 Break

4:15-5:30 Break-out Presentations and Analysis

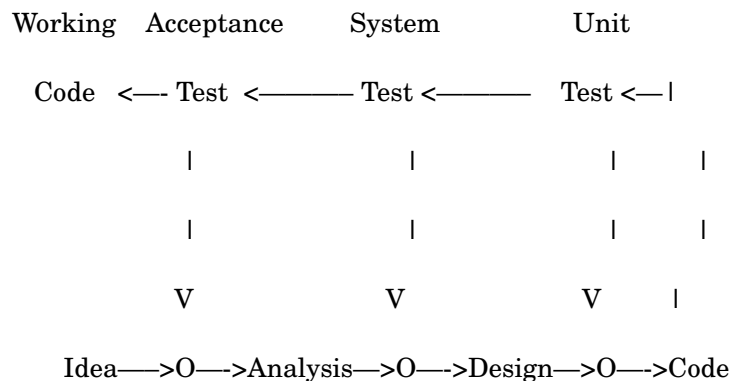
Agile Management Theory - Part II

Five Basic Steps of TOC

1. Identify the system constraint
2. Decide how best to exploit the system constraint
3. Subordinate everything to the decision in step 2
4. Elevate the constraint
5. If steps 1-4 create a new constraint, return to step 2

Software Value Chain

- Value chain is a chain of processes or systems that add value to raw material and create finished product
- Rate of production of finished product is constrained by weakest element in value chain
- Bottleneck, Capacity Constrained Resource (CCR)



Identify and Exploit Constraints

- Capacity of other control mechanisms irrelevant — only one is CCR
- Once identified must decide how to minimize constraining ability
- CCR must always be fully utilized
- Constraints are protected through buffers

Protecting Software Developers

- Protected from idleness by always having a pool of development tasks to be done
- Protected from interruptions by having a good communication mechanisms
- Protected from distractions though quiet work environment

Exploiting (Leveraging) Software Developers

- Exploited through good software development tools
- Exploited through support staff/automated tools for non-productive activities (progress reporting, time tracking)
- Exploited through training
- Exploited through good quality requirements

Subordinating Everything to Constraint

- Hard to do because of cost accounting OE focus
- Drum-Buffer-Rope
- Drum — rate of the CCR which beats out the overall rate of flow
- Buffer — holding place for flow from other places to ensure CCR is never starved
- Rope — flow of inventory
- Parts of system may be idle otherwise inventory builds up
- Software inventory is extremely perishable
- Stale requirements lose the surrounding knowledge base and quickly become useless
- Lost inventory is pure waste

Elevating the Constraint

- Add resources to raise level of drum
- Work overtime — bad in long run as actually reduces output
- Work overtime — bad because not necessarily focusing on true constraints
- Hire better people — difference can be fifty fold
- Surgical teams elevate developer constraint
- Surgical team means top developer actually writing the code so more produced
- Better to add another surgical team than add people to project
- Constraints usually lie in experts not generalists
- Expert consultants elevate constraints

Five Constraints of Software Development and Associated Buffer

1. People - People
2. Schedule - Time
3. Scope (Functionality) — Queue
4. Budget — Money
5. Resources — Resources

Buffering People

- Extra Surgical Teams
- Assume people don't work full 8 hours — 5.5 hours realistic
- Take outage into account — vacation, illness, personal days — 15%
- Size of buffer proportional to uncertainty
- Viewing it as continuous process rather than workday/work week buffers should suffice
- Add people buffer from beginning

Buffering Schedule

- Delivery date can be a hard constraint — e.g. regulatory or customer requirements
- Size of buffer proportional to uncertainty
- Uncertainty related to technology, people, subject matter, architecture, delivery environment, organizational maturity

Certainty	Buffer Size
100%	15%
90%	25-30%
80%	50%
50-70%	100%
<50%	200% (100% min)

Buffering Scope

- Prioritize functions — some may be discarded
- Time-varied priorities — no throughput in early or late delivery
- Scope buffers can only be negotiated when there is customer trust
- Trust comes through understanding, transparency and delivering on promises
- Mistakes are acceptable if they are understood and transparent
- Start by buffering time and people

Buffering Budget

- Proportional to technology and subject matter uncertainty
- Unanticipated work — new tools
- Unanticipated work — more inventory because requirements weren't understood
- Identify external resources
- Use FOSS to lower software licensing costs

Buffering Resources

- Losing a day of work because of resource breakdown is extremely expensive
- Server uptime should be as close to 100% as possible

Four Types of Uncertainty

1. Variation
2. Foreseen Uncertainty
3. Unforeseen Uncertainty
4. Chaos

Variation

- Tasks, activities and work units can be classified through analysis
- Empirical data available for those classifications
- Planning is straightforward
- Degree of uncertainty is bounded
- 2-4 hours to code an HTML page means 3+/-1 hour
- 1000 pages is 3000 hours since over large sample variance averages out
- Endemic to the process
- 100% certainty

Foreseen Uncertainty

- Uncertainty identifiable and understood but may not happen
- Number of pages may change
- Work done is wasted
- Endemic to the process
- 80-90% certainty

Unforeseen Uncertainty

- Pushing the bounds of known technologies
- Extreme innovation
- Firm goals and objectives
- R & D — Research or invention
- 50-70% certainty

Chaos

- Team doesn't know what it's doing or where it's going
- Market not understood
- Customer needs are unclear
- No defined plan
- Constant change
- Software startup
- <50% certainty

Local Safety

- Developers add buffer for each task
- Adding up all the individual tasks leads to ridiculous buffers
- Project may even be canceled due to overestimates or you may lose bid
- Ask developers to estimate by complexity and risk, not effort
- Use codification scheme to do estimates
- Aggregation

Production Metrics

- Inventory level — current location of Investment and progress to becoming Throughput
- Production quantity — Throughput * Lead Time — how long investment committed
- Flow of Value
- Value of working code
- Self-generating based on documentation
- Predictive — Inventory + historical flow gives future information

Measuring Inventory

- Functional Requirements
- Architectural requirements are part of Operating Expense
- Architectural requirements should be set at minimum required
- Architectural requirements that are market differentiating are Inventory
- Tasks are not inventory — tasks are of no interest to the client and not a measure of value
- How inventory is expressed depends on type of method used

Measuring Production Quantity

- Production Quantity (Q) — the output of client valued functions as working code
- Inventory Level (V) =
Approved ideas waiting development +
Functions in active development +
Completed Functions not yet delivered
- Rate of Production (R) — quantity of functional requirements delivered out of the system in a given time period

One Month V(start) = 900
V(end) = 800
Q = 100
R = 100/month

- Goal is to increase R
- Cumulative Flow Diagram — want V moving smoothly through system

Lead Time

- Lead Time — how long it takes a unit of V to move through system from input to output
- Shorter LT means less I
- I is financed so lowering I increases Net Profit
- Customer may be willing to pay more for short LT so it may also increase T

Average Cost Per Function

- ACPF — cost to transform a single unit of inventory through the system

$$\text{ACPF} = \frac{\text{OE/Month}}{\text{Q/Month}}$$

OE = \$1,000,000
Q = 100
ACPF = \$10,000

- Reducing bottlenecks can increase Q without raising OE
- Global optimization not local optimization
- Estimation of future OE

Project Management Models

- ISO/9000-PMI — Lock Scope, Negotiate/Vary Budget, Schedule Delivery Date
- Scope has greatest uncertainty, budget less, date least
- Forces estimate to be accurate when they can't be so add overhead with no result
- Agile Model — Schedule then Budget then Scope
- Requires negotiable scope and scope buffers
- Negotiable scope requires trust
- FDD gives some flexibility on schedule

Three Dimensional Model

- Desire for accurate estimates lead to task planning and effort tracking
- Estimating, measuring and tracking effort doesn't help determine throughput
- One dimensional model is OE focused
- Focus on managing flow through a series of transformations that lead to value
- In cost accounting model, as code passes through phases value is added
- In throughput accounting value decreases since it's becoming stale
- Working code is a liability not an asset unless it's turned into product (T)
- All effort is OE
- Because I depreciates, incentive to lower LT
- Value is recorded only when input is transformed into output
- No need to track tasks or fill out time-sheets
- Management focus leading to global optima

Role of Project Manager

- Old role: create task list, collect estimates, create Gantt charts, maintain charts
- New role: assist flow of value
- Maintain issue log and keep on top of issues before they hamper throughput
- Monitor project buffers
- Keep track of events that might impact delivery

Exercise 6 — Agile Management Theory in Practice

1. You have just been appointed the CIO of Dreeg Inc. It's your first day on the job and you are called into a meeting with the CEO, CFO and VP Human Resources. Apparently the human resources system is in drastic need of an overhaul. due in part to recent new regulations such as HIPAA and the "PATRIOT" act. The CFO comes in with a stack of names of US-managed outsourcing firms. Turning directly to the CEO, she argues that the company can save hundreds of thousands of dollars by outsourcing most of the development work. The VP HR cites privacy and security concerns. The CFO turns to you and says "You can divy up the work so that the parts that get shipped out won't cause any problem, can't you? Your predecessor always managed to do that!" You turn to the CEO and ask for a day to prepare a presentation outlining the pros and cons of in-house development vs. outsourcing. Applying the principles of Agile Management theory, you prepare a presentation for the HR2006 management committee. Be sure to clearly list the pros and cons of each alternative. You also need to include a cost/benefit analysis of the alternative approaches. Assume the management team has no clue about AMT, so you need to explain terms clearly. You can make whatever assumptions you like about constraints in the organization. As part of your plan, be sure to note how you plan to address these constraints.

2. Someone different presents to the class.