# Principles of Product Development Flow

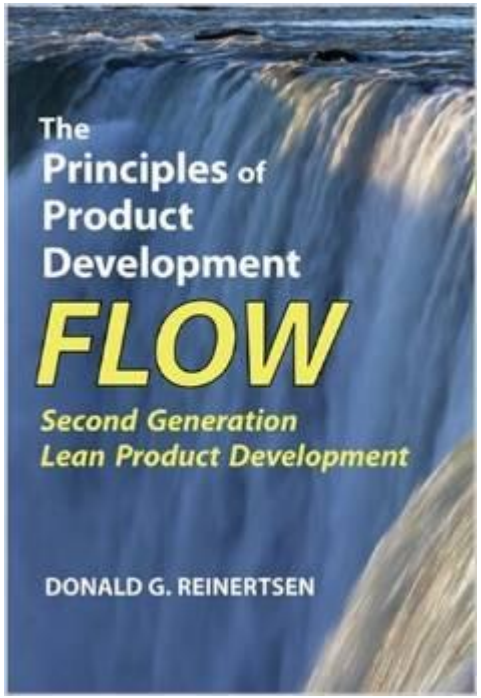
## Part 4: Exploiting Variability

# About Me

- Started programming in 1981
- Owner of Enoki Solutions Inc.
  - Consulting and Software Development
- Exposed to several industries
- Running VanDev since Oct 2010

# Book:
# The Principles of Product Development Flow

- ~$45 on Amazon.ca
- Published in 2009
- Award winning
- Difficult material
- Generally ignored :(
- Awesome IMNSHO



# Why?

- Can Variability lead to bad outcomes? Yes.
- Does it always? No.
- The *amount* of variability is less important than the *cost* of variability

# Risk: Product vs Manufacturing

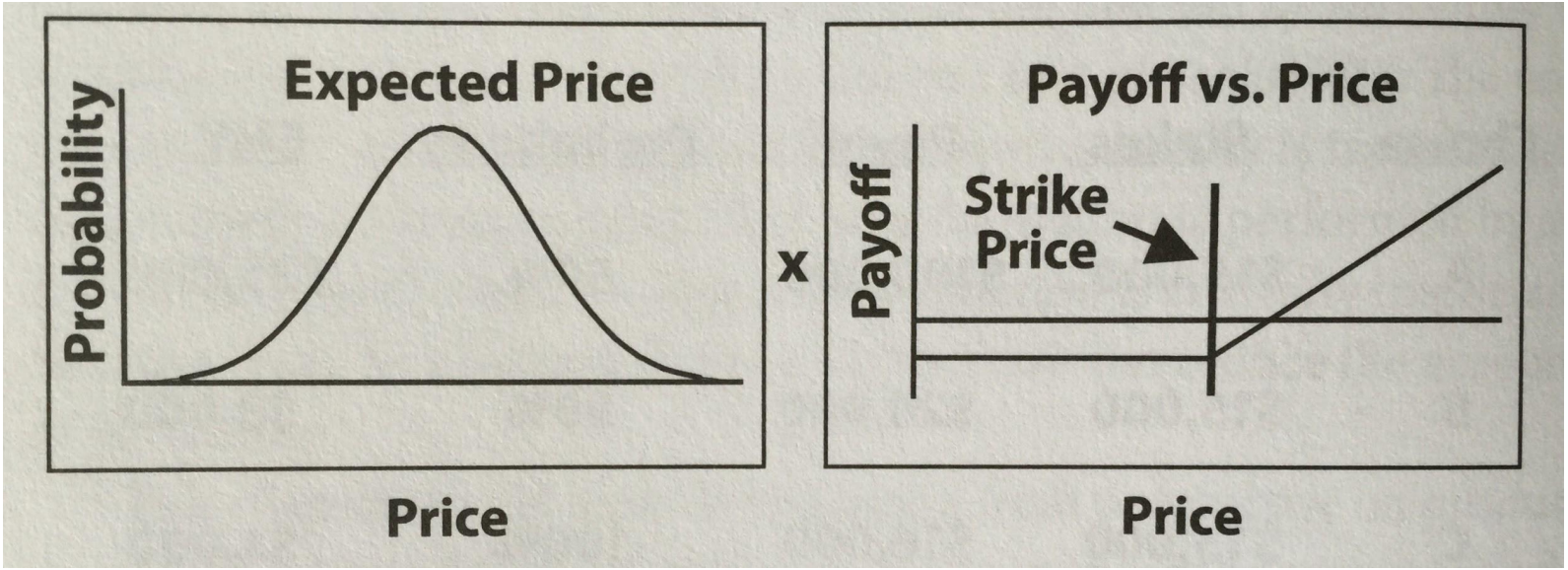
- Product development produces value by producing information
- Manufacturing produces physical products
- Producing the same information gains us no value
- Producing the same physical product gains us value

## **V1 The Principle of Beneficial Variability**: Variability can create economic value.

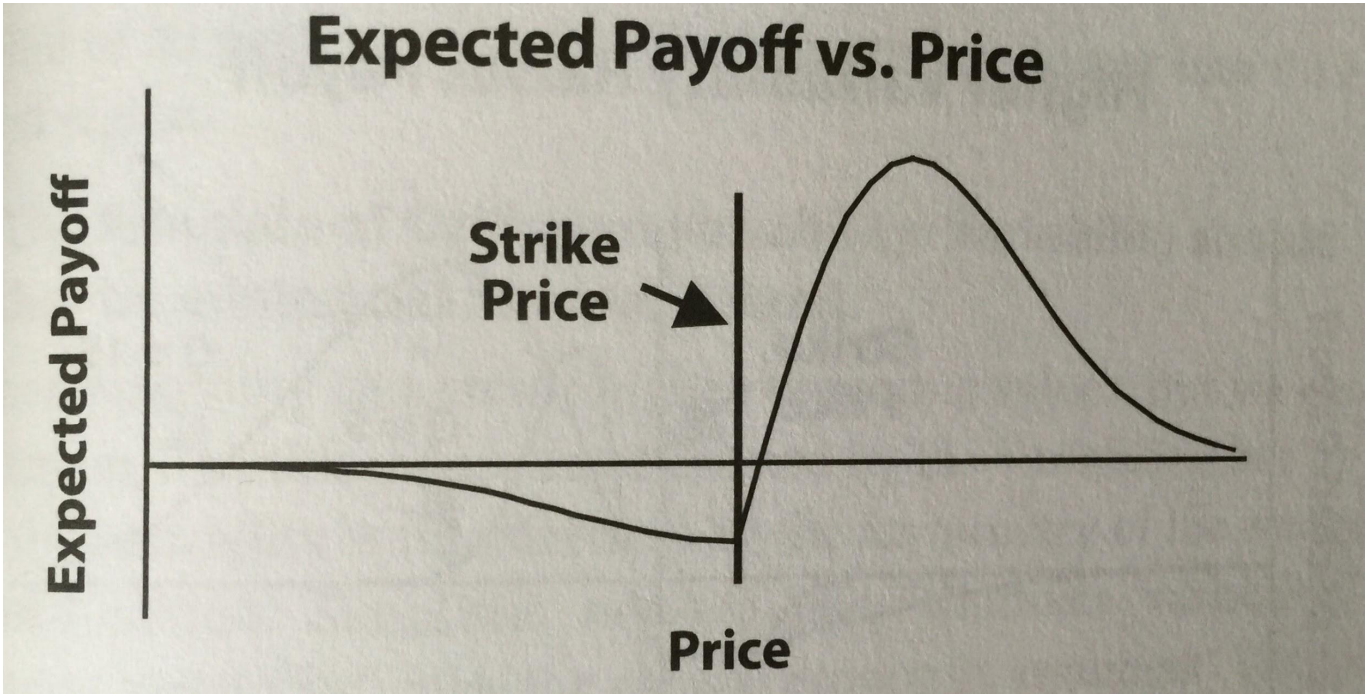
| Choice | Stakes | Payoff | Probability | EMV |
|---|---|---|---|---|
| A | $15,000 | $100,000 | 50% | $35,000 |
| B | $15,000 | $20,000 | 90% | $3,000 |
| C | $15,000 | $16,000 | 100% | $1,000 |

We cannot maximize economic value by eliminating all choices with uncertain outcomes.

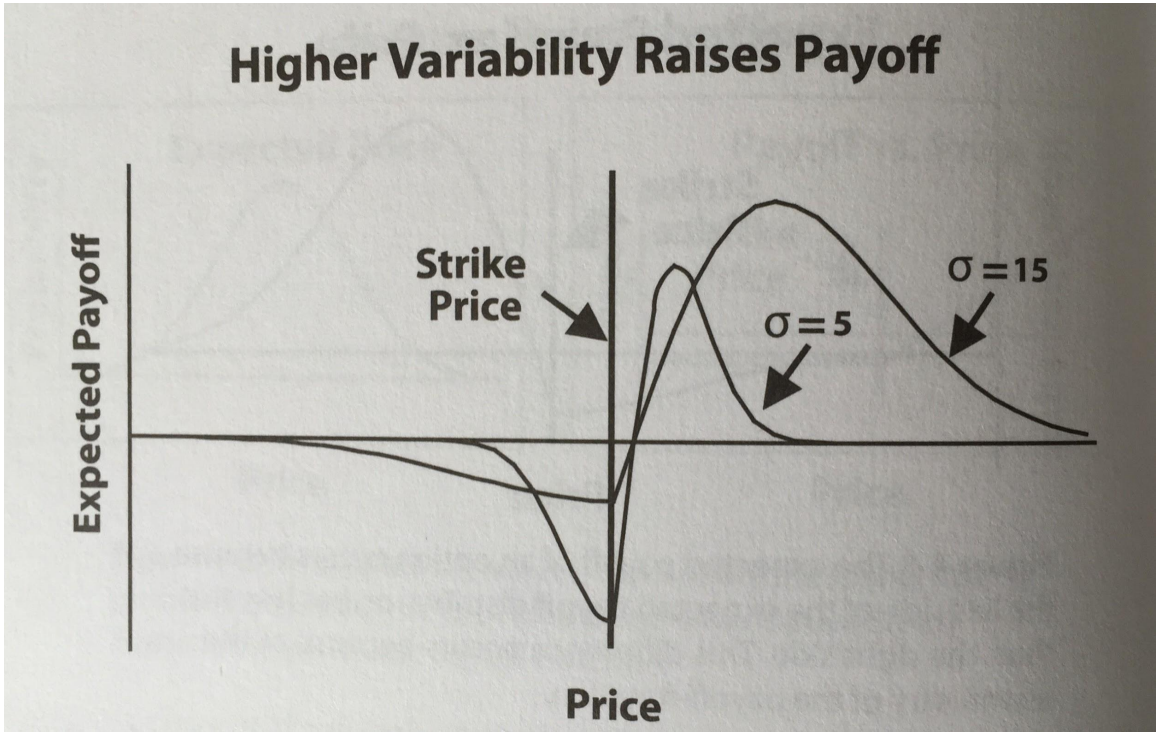
# V2 The Principle of Asymmetric Payoffs: Payoff asymmetries enable variability to create economic value.



# V2 The Principle of Asymmetric Payoffs: Payoff asymmetries enable variability to create economic value.



# V2 The Principle of Asymmetric Payoffs: Payoff asymmetries enable variability to create economic value.



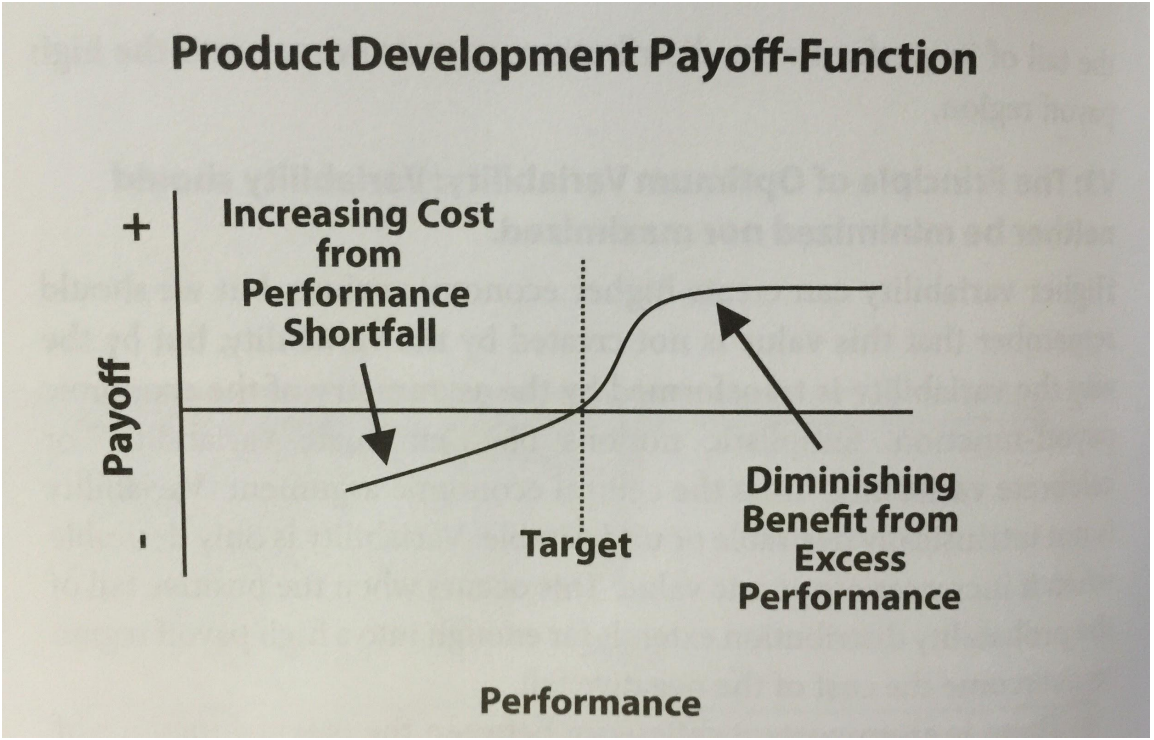
## V2 The Principle of Asymmetric Payoffs: Payoff asymmetries enable variability to create economic value.

- In stocks we want high variability!
- In product development the value of success *can be* much higher than the cost of failure.

## V3 The Principle of Optimal Variability: Variability should neither be minimized nor maximized.

- Simplistic notions like “eliminate variability” or “celebrate variability” miss the central economic argument
- Product development doesn’t have bounded downside

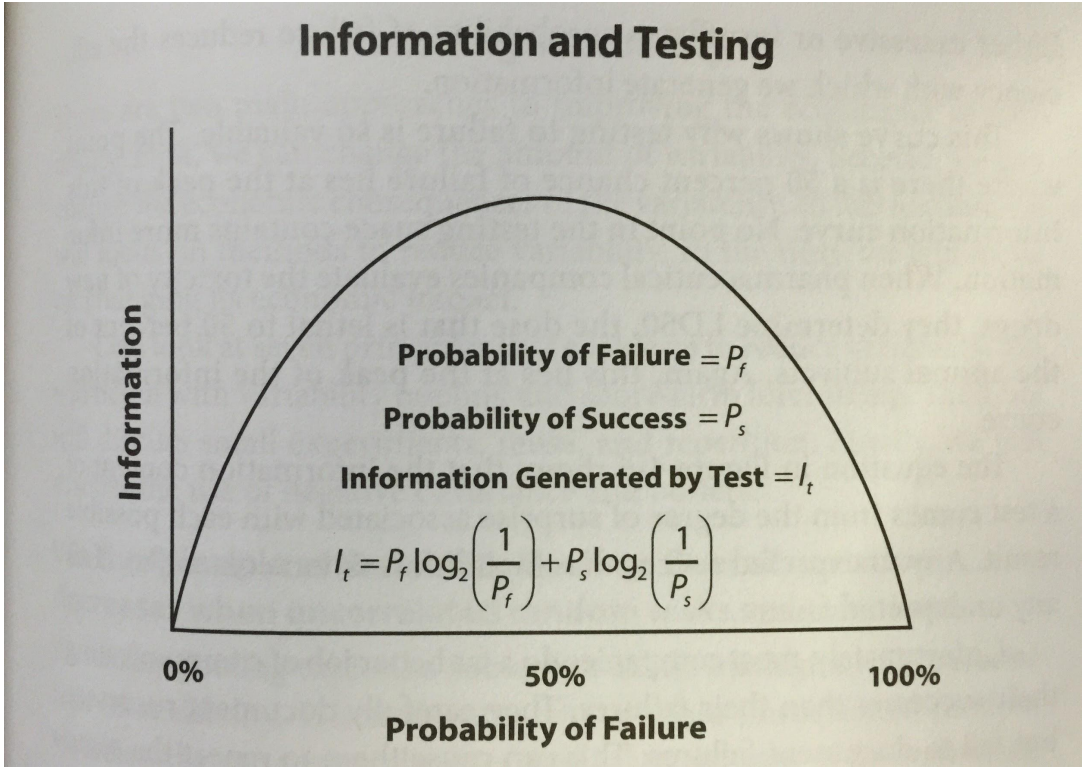
# V3 The Principle of Optimal Variability: Variability should neither be minimized nor maximized.



## V4 The Principle of Optimum Failure Rate: Fifty percent failure rate is usually optimum for generating information.

- The information content of a test comes from the degree of surprise associated with each possible result.
- Repeating a failure gains no information
  - Document your failures and successes
- Resolve uncertainty that is economically valuable

# V4 The Principle of Optimum Failure Rate: Fifty percent failure rate is usually optimum for generating information.



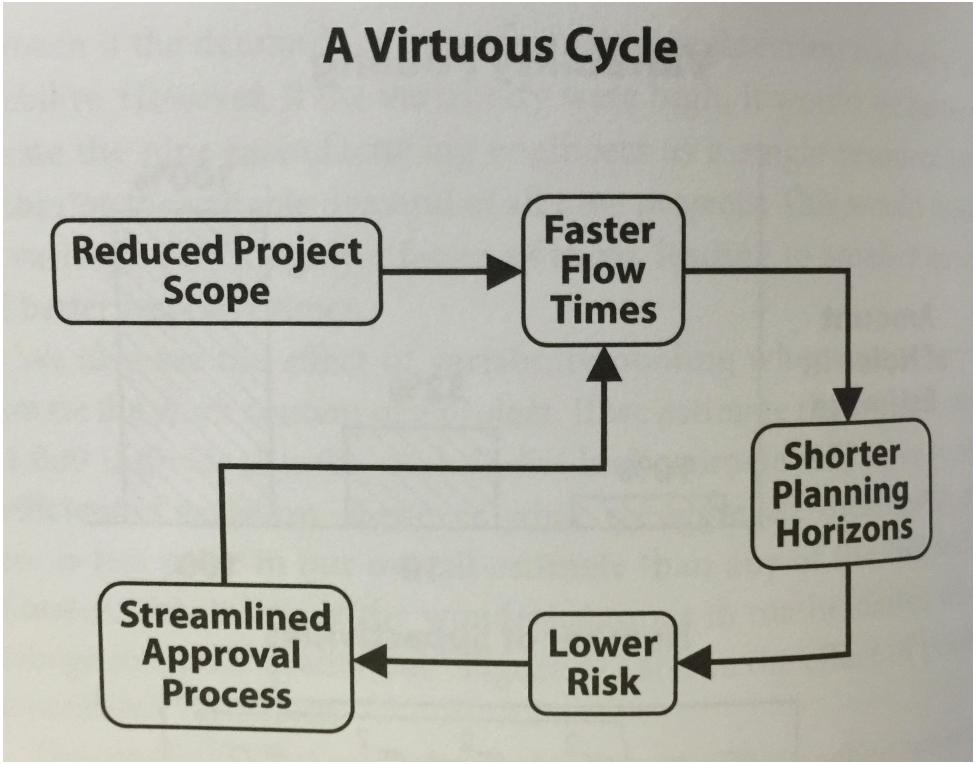
## V5 The Principle of Variability Pooling: Overall variation decreases when uncorrelated random tasks are combined.

- Key word: *uncorrelated*
- Variance of one dice is high
- Sum N dice and it decreases
- Generally I don’t agree that this applies to software as tasks are rarely random and uncorrelated.

## V6 The Principle of Short-Term Forecasting: Forecasting becomes exponentially easier at short time-horizons.

- Given the location of a boat at time $t_0$ and its speed the location at time $t_1$ is a circle proportional to $(t_1-t_0)^2$
- Product development has far more than two degrees of freedom
- Forecasting for 2*t is $2^n$ times harder!

# V6 The Principle of Short-Term Forecasting: Forecasting becomes exponentially easier at short time-horizons.



## V7 The Principle of Small Experiments: Many small experiments produce less variation than one big one.

- 4 coin flips coming up head = 1/16
- Split into 4 tosses
  - ½ we stop and save 3*x cost
  - ¼ we stop and save 2*x cost
  - ⅛ we stop and save 1*x cost
  - 1/16 we succeed at 4*x cost
- x = Original (cost*a)/4
- a = Overhead added by splitting

## V8 The Repetition Principle: Repetition reduces variation.

“As a software development organization moves towards daily build/test cycles, the increasingly routinize their process. If code check-in [or merges] occurs once per year, there is little incentive to invest the one-time effort to structure and optimize the process. If code check-ins occurs daily, we have both the economic incentive and the opportunity to optimize the process.”

## V9 The Reuse Principe: Reuse reduces variability.

- *If* you can reuse something
- *and* the economic benefit of building or using something new is not there.
- Consider:
  - Reusing vacuum tubes vs redesigning for transistors.

## V10 The Principle of Negative Covariance: We can reduce variance by applying a counterbalancing effect.

- Hedge
- Cross train workers so we can react to counter variability actively as it occurs.

## V11 The Buffer Principle: Buffers trade money for variability reduction.

- Padding the schedule
- Converting uncertain earliness to certain lateness
- Generally not a good approach

**V12 The Principle of Variability Consequence: Reducing consequences is usually the best way to reduce the cost of variability.**

- Fail quickly
  → avoid continued costs
- Find/Fix bugs early
  → avoid rework costs
- Require code analysis
  → avoid rework costs
- Reduce task sizes
  → faster delivery
  → faster discovery of defects
  → lower rework costs

## V13 The Nonlinearity Principle: Operate in the linear range of system performance.

- Impact of delay is only linear for short delays
- Projects of anything in the future tend to nonlinear as  the projection moves further into the future
- Avoid assuming anything will stay linear over a long period of time

## V14 The Principle of Variability Substitution: Substitute cheap variability for expensive variability.

- Paying expedite changes for delivery of a part to avoid delay in the project using the part
- Accepting initial defects to make schedule
  - Assuming missing the schedule is more costly than the costs incurred by the defects!

# V15 The Principle of Iteration Speed: It is usually better to improve iteration speed than defect rate.

- I don’t understand his logic†, *but*
- Shorter iterations
  - → fast discovery of defects
  - → fast fixes for defects
  - → less economic damage done by defects

† He seems to disregard the impact of defects on future iterations in his math.

## V16 The Principle of Variability Displacement: Move variability to the process stage where its cost is lowest.

- Pre-production vs Production vs Post-Production
- Hold of starting projects until the pipeline is clear to the end
  - Information decay on early efforts makes them a bad investment
- Too much active → Slow transit time

# Conclusions

- Certain is certainly not your best bet
- Focus on EV, not just risk
- Variability can actually be beneficial
- MVPs should be designed to gain information → 50% failure rate
- Small steps → cheaper failures
  → lower impact of variability

# Q&A