Complexity versus Lean

The Big Showdown

Jurgen Appelo
jurgen@noop.nl
version 1.ks
Goal

To further improve Lean software development by understanding and applying complexity thinking
Agenda

What is complex systems theory?
What is lean software development?

Can we define complexity thinking?
Can we apply complexity thinking?
A new management model

7 principles of Lean software development
5 core practices of Kanban
Various lean practices

Conclusion
Emergent, self-organizing, unpredictable

http://www.flickr.com/photos/judepics/
Complex Systems

“A complex system is a system composed of interconnected parts that as a whole exhibit one or more properties (behavior) not obvious from the properties of the individual parts.”

Sometimes called the sciences of complexity (plural)
General Systems Theory

Study of relationships between elements

Autopoiesis (how a system constructs itself)
Identity (how a system is identifiable)
Homeostasis (how a system remains stable)
Permeability (how a system interacts with its environment)
Cybernetics

Study of regulatory systems

Norbert Wiener
(mathematician)
1894-1964

Goals (the intention of achieving a desired state)
Acting (having an effect on the environment)
Sensing (checking the response of the environment)
Evaluating (comparing current state with system’s goal)
Dynamical Systems Theory
Study of system behavior

Stability (stable states versus unstable states)
Attractors (systems getting sucked into stable states)
Game Theory

Study of co-adapting systems

Competition versus cooperation
Zero sum games versus non-zero sum games
Strategies (including evolutionary stable strategies)

John von Neumann
(mathematician)
1903-1957
Evolutionary Theory

Study of evolving systems

Population (more than one instance)
Replication (mechanism of making new instances)
Variation (differences between instances)
Heredity (differences copied from existing instances)
Selection (environment imposes selective pressure)
Chaos Theory

Study of unpredictable systems

Strange attractors (chaotic behavior)
Sensitivity to initial conditions (butterfly effect)
Fractals (scale-invariance)

Edward Lorenz
(meteorologist)
1917-2008
And more...

Study of all kinds of systems

Dissipative systems (spontaneous pattern-forming)
Cellular automata (complex behavior from simple rules)
Genetic algorithms (adaptive learning)
Social network analysis (propagation of information)
The Body of Knowledge of Systems

Complex systems theory is the study of complex systems using multiple system theories.
Agenda

What is complex systems theory?
What is lean software development?

Can we define complexity thinking?
Can we apply complexity thinking?
A new management model

7 principles of Lean software development
5 core practices of Kanban
Various lean practices

Conclusion
Lean Software Development

Lean manufacturing

14 principles of The Toyota Way

Respect for People & Continuous Improvement

Toyota Production System (TPS)

14 Points for Management (Deming)

Just In Time Production

Lean development is a prescriptive approach to work in social systems

http://www.complexification.net/gallery/machines/treeGarden/
Prescriptive: Lean manufacturing   Lean development

Descriptive: Fundamental Forces of Physics   Complex Systems Theory
Agenda

What is complex systems theory?
What is lean software development?

Can we define complexity thinking?
Can we apply complexity thinking?
A new management model

7 principles of Lean software development
5 core practices of Kanban
Various lean practices

Conclusion
Complexity

“Complexity is that property of a system which makes it difficult to predict its overall behavior, even when given reasonably complete information about its components and their relations.”

“edge of chaos”

“chaordic processes”
The Agreement & Certainty Model

Complex and complicated seen as different *domains*

Simple + Complicated = Ordered;
Complex is between ordered and chaotic
The Cynefin Framework

Complex and complicated seen as different domains

There’s a fifth domain “disorder” in the middle; and a “cliff” between simple and chaotic

http://en.wikipedia.org/wiki/Cynefin
Simplicity: A New Model

**Simple** = structure is easily understandable

**Complicated** = structure is very hard to understand

**Ordered** = behavior is fully predictable

**Complex** = behavior is somewhat predictable

**Chaotic** = behavior is very unpredictable

**Simplification** = making something better understandable

**Linearization** = making something more predictable
Structure-Behavior Model

Complex and complicated seen as different *dimensions*

Unhappy accident: tiger “eats” performer
“Black Swans”: unpredicted big events

Complex system is often predictable, sometimes not
Impact of “unknown unknowns” higher than all else
Risk management deals only with “known unknowns”

• Mirage Casino lost $100 million due to show cancellations
• Airline industry lost billions because of ash cloud from Iceland
• 9/11

http://www.amazon.com/Black-Swan-Improbable-Robustness-Fragility/dp/081297381X/
Happy accident: inkjet printer invention
Serendipity

Accidental discovery while looking for something else
Again: high impact, not predicted

• Inkjet printer invented when putting soldering iron on pen
• Viagra invented by accident (by Pfizer)
• America discovered by accident

http://www.amazon.com/Black-Swan-Improbable-Robustness-Fragility/dp/081297381X/
Distinguishing Cause and Effect
Non-Linear Behavior

Sometimes hard to distinguish cause and effect
Sometimes many causes for one effect

- Low quality from high pressure, or high pressure from low quality?
- Bad atmosphere because colleagues are cranky, or colleagues are cranky because of bad atmosphere?

http://www.amazon.com/Black-Swan-Improbable-Robustness-Fragility/dp/081297381X/
Agenda

What is complex systems theory?
What is lean software development?

Can we define complexity thinking?
Can we apply complexity thinking?
A new management model

7 principles of Lean software development
5 core practices of Kanban
Various lean practices

Conclusion
System Dynamics

Study of non-linear behavior of systems

Circular feedback loops and time-delayed relationships
Analysis through simulations and calculations

Jay Wright Forrester
(computer engineer)
1918-
Systems Thinking

Approach to problem solving

“Problems” are part of a system
View systems in a holistic manner
Not a science, but a “frame of mind”

Peter Michael Senge
(social scientist)
1947-
Some Criticism

“The strength of systems thinking is its recognition that human systems are messy, they frequently need focus and alignment; its weakness is that it assumes that the design of that focus and alignment is a top down objective based process. [...] The ambiguity of human systems is recognized, but the basic concept of central control or planning remains at the heart.”

Multi-ontology sense-making - David Snowden (2005)

Some Criticism

“Systems thinking contains a fundamental difficulty right at its roots. This is to regard human interaction as a system. This assumption leads to thinking about that interaction as something about which another human standing outside it makes choices.”

*Complexity and Management* – Ralph Stacey (2000)
“Traditional” Systems Thinking

observer

system

Complexity Thinking

observers

system

Example

“The purpose of a business is to find/satisfy customers (produce customer value).”
Example

“The purpose of a business is to find/satisfy customers (produce customer value).”
Example

“The purpose of a business is to find/satisfy customers (produce customer value).”

“The purpose of a business is whatever emerges from the interaction of stakeholders.”
Complexity Thinking

Don’t separate the designers from the system
Don’t ignore the human part (social complexity)
Don’t ignore the unknown unknowns
Don’t rely (too much) on linear cause and effect

Complexity Thinking = Systems Thinking++
Agenda

What is complex systems theory?
What is lean software development?

Can we define complexity thinking?
Can we apply complexity thinking?

A new management model

7 principles of Lean software development
5 core practices of Kanban
Various lean practices

Conclusion
The Management 3.0 Model

Six organizational views based on complexity thinking
View #1: Energize People

People are the most important parts of an organization and managers must do all they can to keep people active, creative, and motivated.
View #2: **Empower Teams**

Teams can self-organize, and this requires empowerment, authorization, and trust from management.
View #3: Align Constraints

Self-organization can lead to anything, and it’s therefore necessary to protect people and shared resources, and to give people a clear purpose and defined goals.
View #4: Develop Competence

Teams cannot achieve these goals if team members aren’t capable enough, and managers must therefore contribute to the development of competence.
View #5: Grow Structure

Many teams operate within the context of a complex organization, and thus it is important to consider structures that enhance communication.
View #6: Improve Everything

People, teams, and organizations need to improve continuously to defer failure for as long as possible.
Agenda

What is complex systems theory?
What is lean software development?

Can we define complexity thinking?
Can we apply complexity thinking?
A new management model

7 principles of Lean software development
5 core practices of Kanban
Various lean practices

Conclusion
Principle 1: Eliminate Waste

http://www.flickr.com/photos/braydawg/202403468/
Principle 1: *Eliminate Waste*

Yes, but...

**Junk DNA** (98%) enables innovation and resilience

You cannot measure the *unexpected value of waste*

And thus:

There can be value in having waste “lying around”

If the cost of waste is low, maybe you should keep it
Principle 2: Build Quality In

http://www.flickr.com/photos/extranoise/276297674/
Principle 2: Build Quality In

Yes, but...

Preventing errors inhibits learning from errors
Limiting ways of usage limits innovation
You cannot predict the value of doing things “wrong”

And thus:
There can be value in being ambiguous & inexplicit
Principle 3: Create Knowledge

http://www.flickr.com/photos/butterflysha/135659489/
Principle 3: Create Knowledge

Yes, but...

Develop Competence = skill * discipline * knowledge * social connectivity

And connectivity has more effect than knowledge


And thus:

Competence in the system is more than knowledge
Principle 4: Defer Commitment

http://www.flickr.com/photos/zoutedrop/2317065892/
Principle 4: Defer Commitment

Yes, but...

Committing early can be motivating.

Committing early changes risks and opportunities.

You cannot predict the results of these changes.

And thus:

There is (sometimes) value in making early choices.
Principle 5: Deliver Fast

http://www.flickr.com/photos/curns/4237949343/
Principle 5: Deliver Fast

Yes, but...

This assumes adaptation as a survival strategy
Humans are successful thanks to consciousness
Also called an “anticipation device” (Daniel Dennett)
Anticipation can (sometimes) outsmart adaptation

And thus:
Think (briefly), then deliver fast
Principle 6: Respect People

http://www.flickr.com/photos/fhashemi/44551132/
Principle 6: Respect People

Yes, but...

Energize People = trust * respect * motivation * diversity * creativity

Respect is insufficient to instill a “need” for work

And thus:
People in the system must be energized
Principle 7: Optimize the Whole

http://www.flickr.com/photos/flatbag74/2884704365/
Principle 7: Optimize the Whole

Yes, but...

Cross-functional teams can be sub-optimizing too

“Optimize the whole” invites top-down control

A complex system finds its own global optimum through local optimizations and global dependencies

And thus:

Create locally optimizing and interdependent teams
“Missing” in Lean Software Development

Not covered
“Missing” in Lean Software Development
Not covered
No guidance on structuring organizations
No explicit choice for value networks over hierarchies
That’s why Lean is abused in top-down “lean & mean”
Agenda

What is complex systems theory?
What is lean software development?

Can we define complexity thinking?
Can we apply complexity thinking?
A new management model

7 principles of Lean software development
5 core practices of Kanban
Various lean practices

Conclusion
1: Visualize the workflow

http://www.flickr.com/photos/audreyjm529/235458062/
1: Visualize the workflow

Yes, but...

“Value stream” and “value chains” suggest a linear flow of value

But an organization has **many stakeholders**, who all want to get value out of their collaboration

And thus...

Visualize multiple workflows
2: Limit work in progress (WIP)

http://www.flickr.com/photos/blumpy/325853852/
2: Limit work in progress (WIP)

Yes, but...
A book author has an entire book in progress
Limited WIP is just one example of a constraint

And thus:
A system will self-organize around its constraints
Choose constraints to match the workflow
3: Measure and manage flow
3: Measure and manage flow

Yes, but...

Leads to sub-optimization when only flow to customers is considered

And thus:

Measure and manage flow to all stakeholders
(*customers, suppliers, employees, shareholders, ...*)
4: Make process policies explicit

http://www.flickr.com/photos/moonlightbulb/3323103034/
4: Make process policies explicit

Yes, but...

*Genetic algorithms*: a variety of conflicting rules
Learning made possible through “credit assignment”

And thus:
Experiment with rules
5: Use models to suggest improvements

http://www.flickr.com/photos/jurvetson/447302275/
5: Use models to suggest improvements

Yes, but...

All models are wrong, some are useful

Beware of “scientific approach” to workflows
Banks and casinos also have “scientific approaches”
Yet, they have been unpleasantly surprised...

And thus:
Use complexity thinking to suggest models
“A Model for Creating a Kanban System”

http://availagility.co.uk/2010/10/11/a-model-for-creating-a-kanban-system/
“A Model for Creating a Kanban System”

http://availagility.co.uk/2010/10/11/a-model-for-creating-a-kanban-system/
“Missing” in Kanban
Not covered
Agenda

What is complex systems theory?
What is lean software development?

Can we define complexity thinking?
Can we apply complexity thinking?
A new management model

7 principles of Lean software development
5 core practices of Kanban
Various lean practices

Conclusion
Root Cause Analysis

Suggests there is a root cause
But often there are non-linear relationships
Unclear what is cause and what is effect
Kaizen

Suggests gradual linear process improvement
But systems can get stuck in a local optimum on the fitness landscape

Sometimes need for radical change (kaikaku)
Agenda

What is complex systems theory?
What is lean software development?

Can we define complexity thinking?
Can we apply complexity thinking?
A new management model

7 principles of Lean software development
5 core practices of Kanban
Various lean practices

Conclusion
The 1st "danger" of Lean misunderstanding

Relying on data, ignoring Black Swans

<table>
<thead>
<tr>
<th>APPLICATION</th>
<th>Simple payoffs</th>
<th>Complex payoffs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domain</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Distribution 1</td>
<td>Extremely robust to Black Swans</td>
<td>Quite robust to Black Swans</td>
</tr>
<tr>
<td>Distribution 2</td>
<td>Quite robust to Black Swans</td>
<td>LIMITS of Statistics – extreme fragility to Black Swans</td>
</tr>
</tbody>
</table>

http://www.edge.org/3rd_culture/talebo8/talebo8_index.html
The 2nd “danger” of Lean misunderstanding

Relying on cause and effect, ignoring non-linearity
The 3th “danger” of Lean misunderstanding

Ignoring value networks, multiple stakeholders

http://www.edge.org/3rd_culture/taleb08/taleb08_index.html
Lean is GREAT!

(really, it is)

But let’s not stop thinking.
the end
slideshare.net/jurgenappelo

@jurgenappelo (twitter)

noop.nl (blog)
Q & A
This presentation was inspired by the works of many people, and I cannot possibly list them all. Though I did my very best to attribute all authors of texts and images, and to recognize any copyrights, if you think that anything in this presentation should be changed, added or removed, please contact me at jurgen@noop.nl.

You are free:

- to Share — to copy, distribute and transmit the work

Under the following conditions:

- **Attribution.** You must attribute the work in the manner specified by the author or licensor (but not in any way that suggests that they endorse you or your use of the work).
- **No Derivative Works.** You may not alter, transform, or build upon this work.

- For any reuse or distribution, you must make clear to others the license terms of this work. The best way to do this is with a link to this web page.
- Any of the above conditions can be waived if you get permission from the copyright holder.
- Nothing in this license impairs or restricts the author's moral rights.

http://creativecommons.org/licenses/by-nd/3.0/