HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES

h_da

Agile Software Development

Part 2: Towards Adaptive Software Development

Prof. Dr. A. del Pino

Towards Adaptive Software Development

How to deal with complexity ?

Towards Adaptive Software Development

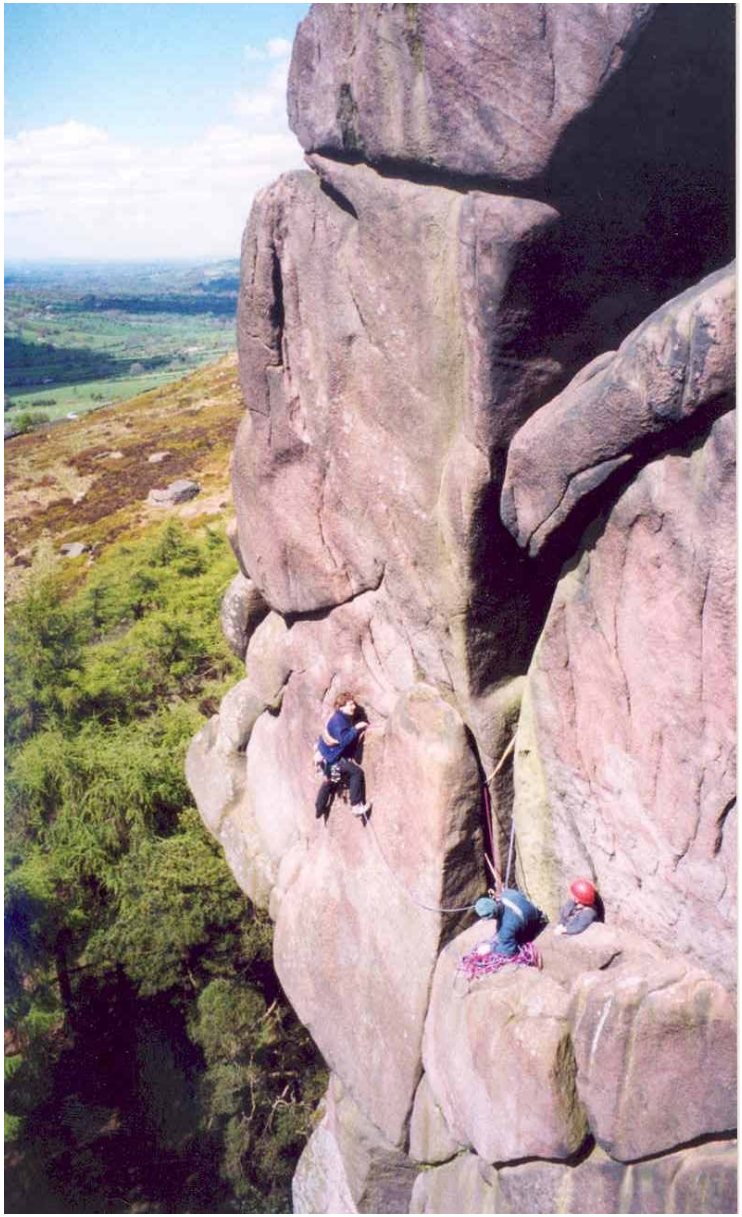


Image source: Wikipedia

There are all kinds of climbing environments, ranging from the afternoon recreational outing to those stretching the limits of human capability. Software projects are the same – from those any reasonably skilled team can complete to those only a few world-class organizations should attempt.

Source: J. Highsmith. *Adaptive Software Development*. Dorset House Publishing, 2000, pg. 3

Towards Adaptive Software Development

Monumental vs. Accidental Software Development

Image source: J. Highsmith. *Adaptive Software Development*. Dorset House Publishing, 2000, pg. 5

Towards Adaptive Software Development

A third alternative

Between chaos and orderliness lies the realm of the complex – a transition zone where imposed order is replaced by *emergent* order, where neither Monumental not Accidental approaches have sufficient diversity to succeed.

Emergent order is a property of complex adaptive systems, generally associated with living entities and their relationships, whose principles help us understand fields as diverse as ecology and organizational management.

Source: J. Highsmith. *Adaptive Software Development*. Dorset House Publishing, 2000, pg. 3

Towards Adaptive Software Development

Complex Adaptive Systems (CAS)

> CAS are, without an exception, made of large numbers of active elements that [...] are diverse in both form and capability.

Source: John H. Holland. *Hidden Order. How Adaptation Builds Complexity*. Basic Books, 1995, pg. 6

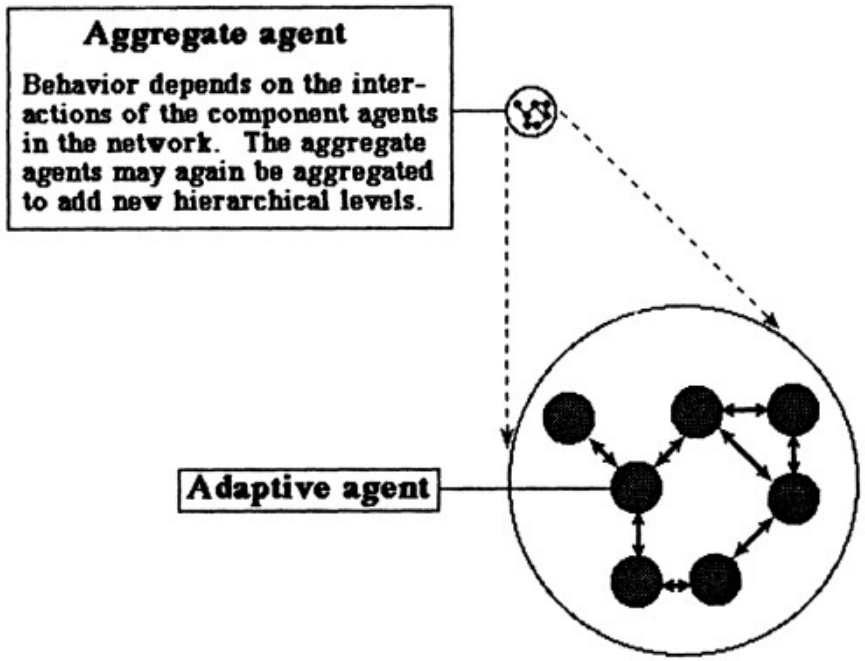


Image source: John H. Holland. *Hidden Order. How Adaptation Builds Complexity*. Basic Books, 1995, pg. 6

Towards Adaptive Software Development

Adaptive Agents

An *adaptive agent* is the abstraction of an active element in a CAS.

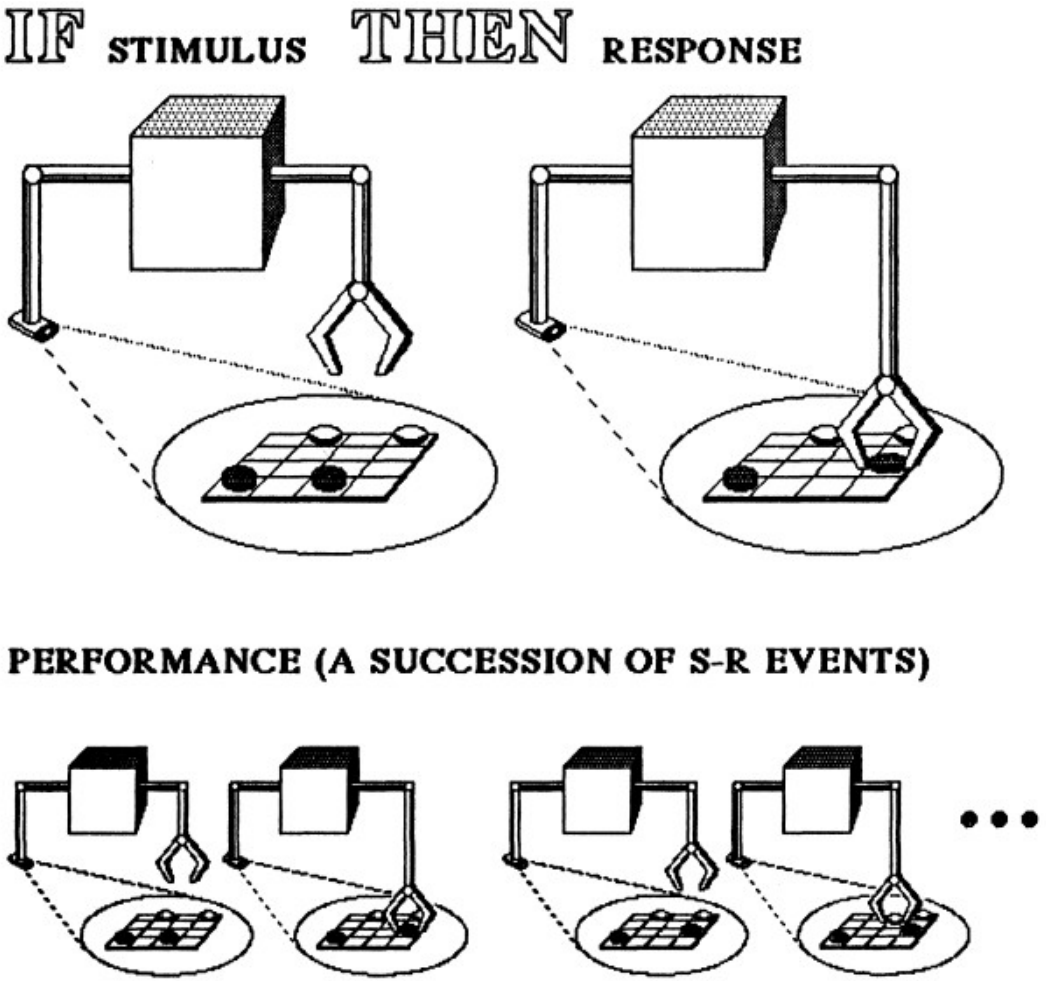


Image source: John H. Holland. *Hidden Order. How Adaptation Builds Complexity*. Basic Books, 1995, pg. 7

Towards Adaptive Software Development

Adaptation

> Adaptation, in biological usage, is the process whereby an organism fits itself to its environment. [...] Here we expand the term's range to include learning and related prcesses.

Source: John H. Holland. *Hidden Order. How Adaptation Builds Complexity*. Basic Books, 1995, pg. 6

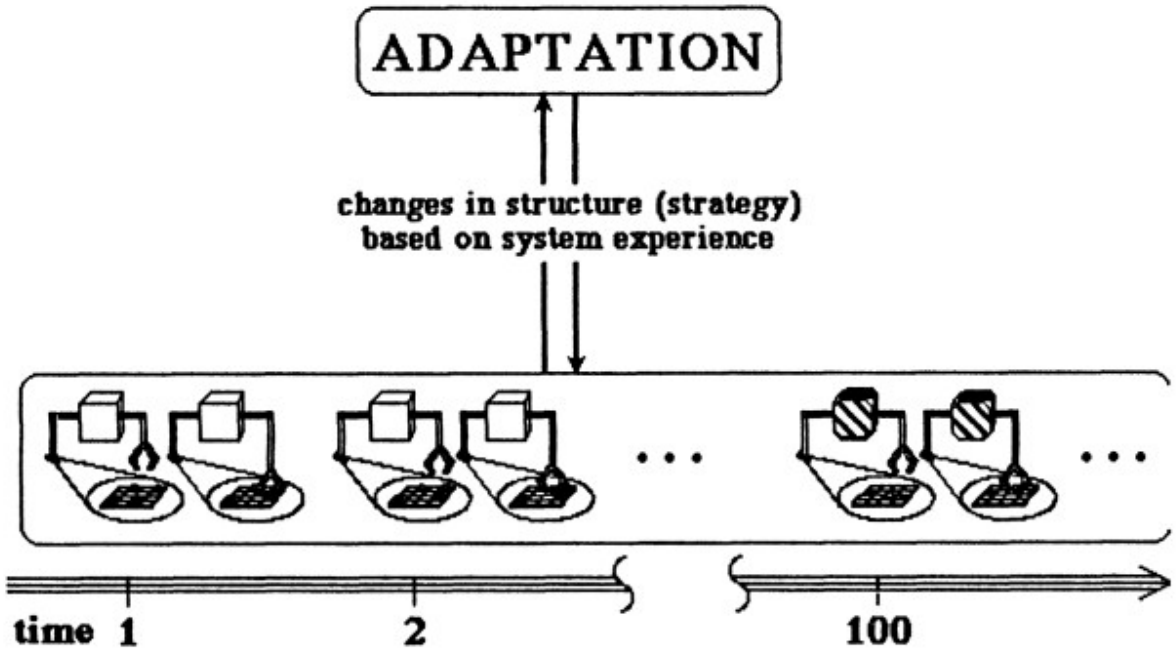


Image source: John H. Holland. *Hidden Order. How Adaptation Builds Complexity*. Basic Books, 1995, pg. 9

Towards Adaptive Software Development

Seven Basics of CAS – Aggregation

> [Aggregation] concerns the emergence of complex large-scale behaviors from the aggregate interactions of less complex agents.

Source: John H. Holland. *Hidden Order. How Adaptation Builds Complexity*. Basic Books, 1995, pg. 6

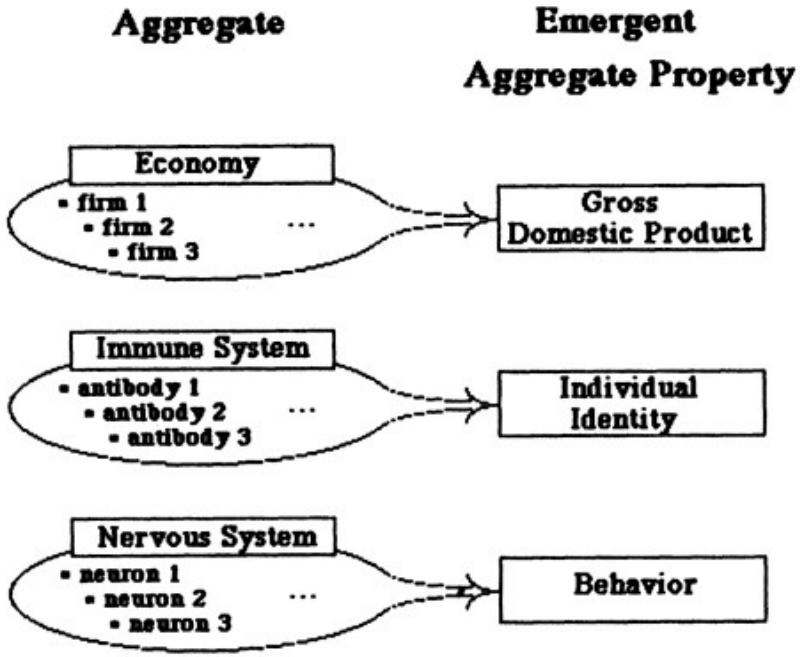


Image source: John H. Holland. *Hidden Order. How Adaptation Builds Complexity*. Basic Books, 1995, pg. 12

Towards Adaptive Software Development

Seven Basics of CAS – Tagging

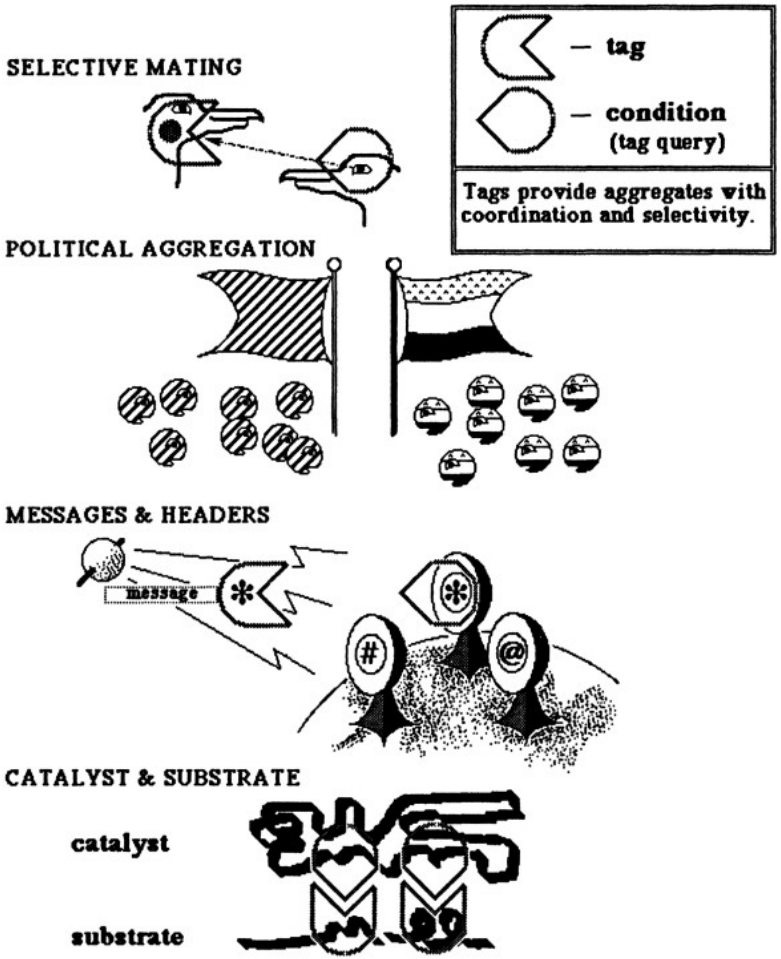


Image source: John H. Holland. *Hidden Order. How Adaptation Builds Complexity*. Basic Books, 1995, pg. 12

[Tags] allow agents to select among agents or objects that would otherwise be indistinguishable. Well-established tag-based interactions provide a sound basis for filtering, specialization, and cooperation. This, in turn, leads to the emergence of meta-agents and organizations that persist even though their components are continually changing.

Source: John H. Holland. *Hidden Order. How Adaptation Builds Complexity*. Basic Books, 1995, pg. 14

Towards Adaptive Software Development

Seven Basics of CAS – Nonlinearity

> Nonlinear interactions almost always make the behavior of the aggregate more complicated that would be predicted by summing or averaging.
>
> Source: John H. Holland. *Hidden Order. How Adaptation Builds Complexity*. Basic Books, 1995, pg. 23

Examples

- Predator/prey models.
- Brooke's Law – Adding people late to a project makes it later.

Towards Adaptive Software Development

Seven Basics of CAS – Flows

> [..] we think of flows over a network of nodes and connectors. [...] In general terms, the nodes are processors – agents – and the connectors designate the possible interactions. In cas the flows through these networks vary over time [...] as the agents adapt or fail to adapt. Thus neither the flows nor the networks are fixed in time. They are patterns that reflect changing adaptations as time elapses and experience accumulates.

Source: John H. Holland. *Hidden Order. How Adaptation Builds Complexity*. Basic Books, 1995, pg. 23

Examples of flows in a software development project ?

Towards Adaptive Software Development

Seven Basics of CAS – Diversity

> [...] diversity is neither accidental nor random. The persistence of any individual agent, whether organism, neuron, or firm, depends on the context provided by the other agents. Roughly, each kind of agent fills a niche that is defined by the interactions centering on that agent. If we remove one kind of agent from the system, creating a "hole," the system typically responds with a cascade of adaptations resulting in a new agent that "fills the hole."

Source: John H. Holland. *Hidden Order. How Adaptation Builds Complexity*. Basic Books, 1995, pg. 27

Examples of diversity in software development ?

Towards Adaptive Software Development

Seven Basics of CAS – Internal models

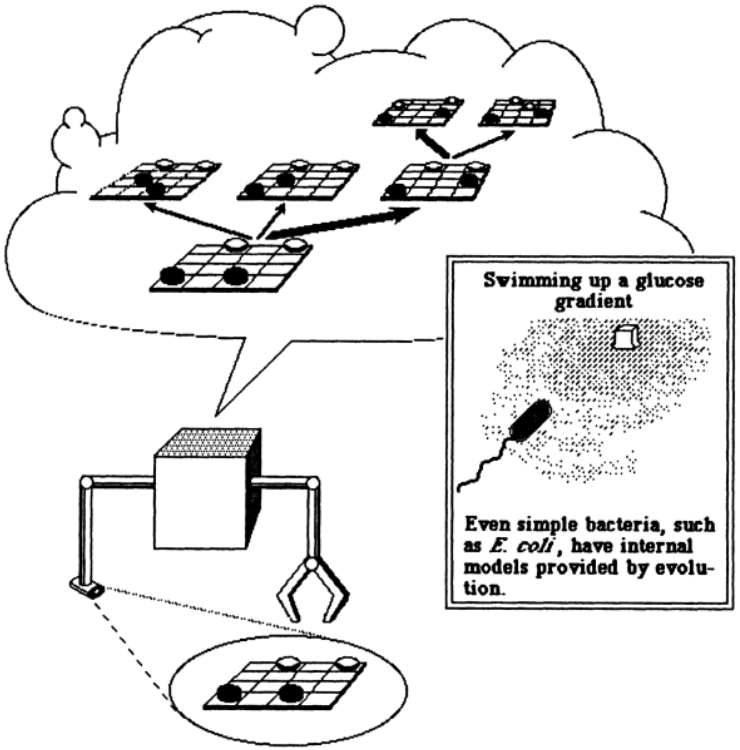


Image source: John H. Holland. *Hidden Order. How Adaptation Builds Complexity*. Basic Books, 1995, pg. 32

[...] the agent must select patterns in the torrent of input it receives and then must convert those patterns into changes in its internal structure. Finally, the changes in structure, the model, must enable the agent to anticipate the consequences that follow when that pattern (or one like it) is again encountered.

Source: John H. Holland. *Hidden Order. How Adaptation Builds Complexity*. Basic Books, 1995, pg. 31

Towards Adaptive Software Development

Seven Basics of CAS – Building blocks

We gain experience through repeated use of building blocks, even though they may never twice appear in exactly the same combination. [...] This use of building blocks to generate internal models is a pervasive feature of complex adaptive systems.

Source: John H. Holland. *Hidden Order. How Adaptation Builds Complexity*. Basic Books, 1995, pg. 34

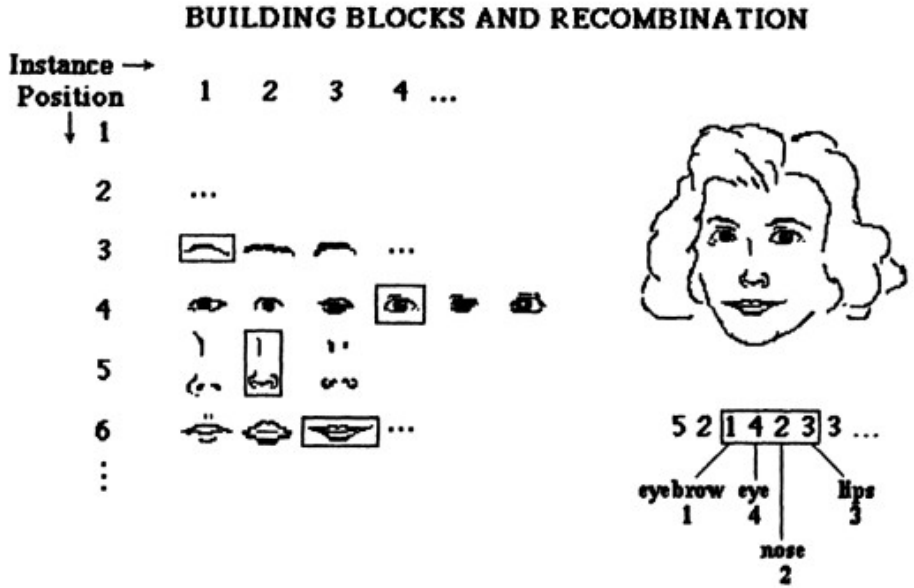


Image source: John H. Holland. *Hidden Order. How Adaptation Builds Complexity*. Basic Books, 1995, pg. 36

Towards Adaptive Software Development

Machines vs. Organisms

Complex adaptive systems, self-organization, emergence, and non-determinism are all scientific, descriptive, but somewhat sterile terms. What they mean is that organizations – project teams – are not machines, but living organisms. [...]

Approaching software development as an adaptive process, and viewing the project team itself as a living organism rather than as an impersonal machine, provides a better model for managing extreme software projects.

Source: J. Highsmith. *Adaptive Software Development*. Dorset House Publishing, 2000, pg. 11

Towards Adaptive Software Development

Traditional Business - Diminishing Returns

Model developed by the economist *Alfred Marshall* (1842-1924) and others:

> Products or companies that get ahead in a market eventually run into limitations, so that a predictable equilibrium of prices and market shares is reached.
>
> Source: W. Brian Arthur. *Increasing Returns and the New World of Business*. Harvard Business Review, July-August 1996

Examples

- Manufacturing
- Bulk-material processing
- Farming industries

Towards Adaptive Software Development

The World of Increasing Returns

Example: The market for operating systems for PCs in the 1980s:

> CP/M was first in the market and by 1979 was well established. The Mac arrived later, but it was wonderfully easy to use. DOS was born when Microsoft locked up a deal in 1980 to supply an operating system for the IBM PC. For a year or two, it was by no means clear which system would prevail.

Source: W. Brian Arthur. *Increasing Returns and the New World of Business*. Harvard Business Review, July-August 1996

Towards Adaptive Software Development

Properties of Increasing Returns

> [M]arket instability (the market tilts to favor a product that gets ahead), multiple potential outcomes (under different events in history, different operating systems could have won), unpredictability, the ability to lock in a market, the possible predominance of an inferior product, and fat profits for the winner.

Source: W. Brian Arthur. *Increasing Returns and the New World of Business*. Harvard Business Review, July-August 1996

Towards Adaptive Software Development

Two economic worlds

> [A] bulk-production world yielding products that are essentially congealed resources with a little knowledge and operating according to Marshall's principles of diminishing returns, and a knowledge-based part of the economy yielding products that essentially are congealed knowledge with a little resources and operating under increasing returns.
>
> Source: W. Brian Arthur. *Increasing Returns and the New World of Business*. Harvard Business Review, July-August 1996

- Not always a clean split, e.g. HP.
- Different economics, different approach to management.

Towards Adaptive Software Development

Competition in the decreasing returns world

> Production tends to be repetitive – much the same from day to day or even year to year. Competing therefore means keeping product flowing, trying to improve quality, getting costs down. [...] It favors an environment free of surprises or glitches – an environment characterized by control and planning. [...] And so, Marshall's world tends to be one that favors hierarchy, planning, and controls. Above all, it is a world of optimization.

Source: W. Brian Arthur. *Increasing Returns and the New World of Business*. Harvard Business Review, July-August 1996

Towards Adaptive Software Development

Competition in the world of increasing returns

By contrast, the style of competition in the increasing returns arena is more like gambling. [...] It is casino gambling, where part of the game is to choose which games to play, as well as playing them with skill. [...] What counts to some degree – but only to some degree – is technical expertise, deep pockets, will, and courage. Above all, the rewards go to the players who are first to make sense of the new games looming out of the technological fog, to see their shape, to cognize them.

Source: W. Brian Arthur. *Increasing Returns and the New World of Business*. Harvard Business Review, July-August 1996

Towards Adaptive Software Development

Back to Adaptation

> You cannot optimize in the casino of increasing-returns games. [...] What you *can* do is adapt. Adaptation [...] means watching for the next wave that is coming, figuring out what shape it will take, and positioning the company to take advantage of it. Adaptation is what drives increasing-returns businesses, not optimization.

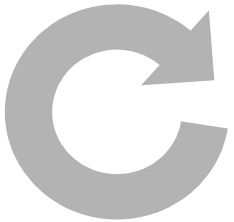
Source: W. Brian Arthur. *Increasing Returns and the New World of Business*. Harvard Business Review, July-August 1996

Towards Adaptive Software Development

Conclusion

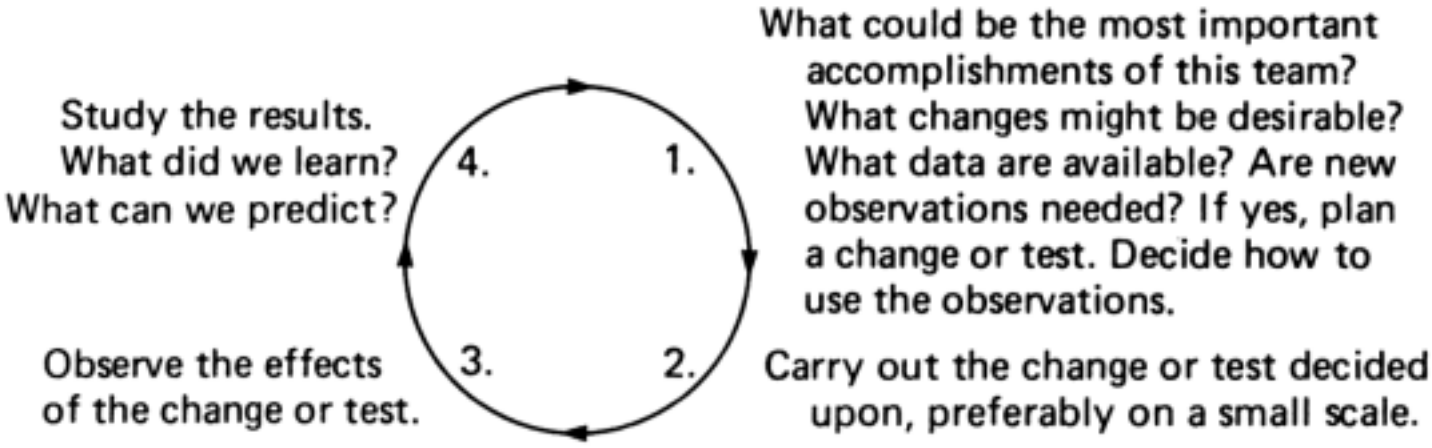
> In complex environments, adaptation is significantly more important than optimization. [...] Optimization works in a complicated world, adaptation works in a complex one.

Source: J. Highsmith. *Adaptive Software Development*. Dorset House Publishing, 2000, pg. 12



Towards Adaptive Software Development

The Deming/Shewhart cycle, a.k.a. Plan Do Check Act (PDCA)



Step 5. Repeat Step 1, with knowledge accumulated.
Step 6. Repeat Step 2, and onward.

Image source: W. Edwards Deming. *Out of the Crisis.* MIT Press, 2000, pg. 88

> I called [the cycle] in Japan in 1950 and onward the Shewhart cycle. It went into immediate use in Japan under the name of the Deming cycle, and so it has been called there ever since.

Source: W. Edwards Deming. *Out of the Crisis*. MIT Press, 2000, pg. 88

Towards Adaptive Software Development

Tom Gilb, 1970s - Evolution

„Evolution“ is a technique for producing the appearance of stability. A complex system will be most successful if it is implemented in small steps and if each step has a clear measure of successful achievement as well as a „retreat“ possibility to a previous successfuly step upon failure.

Source: T. Gilb. *Software Metrics*. Winthrop, 1977

Towards Adaptive Software Development

Barry Boehm, 1980s – Spiral Model

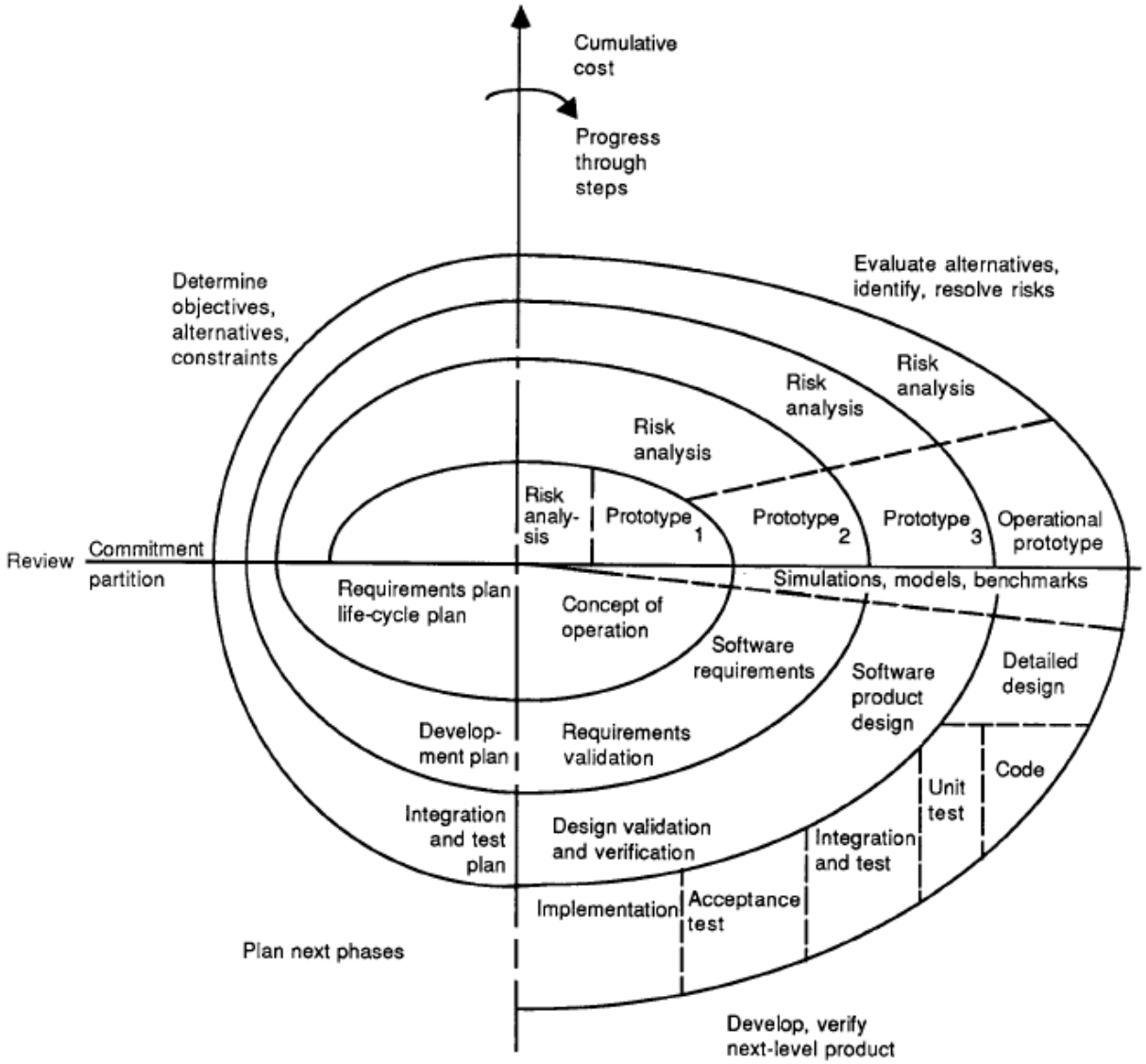


Image source: Barry W. Boehm. *A Spiral Model of Software Development and Enhancement*. IEEE Computer, Vol. 21, No. 5, May 1988

Towards Adaptive Software Development

Further Reading

- J. Highsmith. *Adaptive Software Development*. Dorset House Publishing, 2000
- John H. Holland. *Hidden Order. How Adaptation Builds Complexity*. Basic Books, 1995
- W. Brian Arthur. *Increasing Returns and the New World of Business*. Harvard Business Review, July-August 1996
- C. Larman, V. R. Basili. *Iterative and Incremental Development: A Brief History*. IEEE Computer, Vol. 36, No. 6, June 2003