Alfred Holl

The role of cooperative cyclic knowledge gain in IS anti-aging

1. IS modification as process of cooperative cyclic knowledge gain

Organizations (profit and non-profit) are open, complex

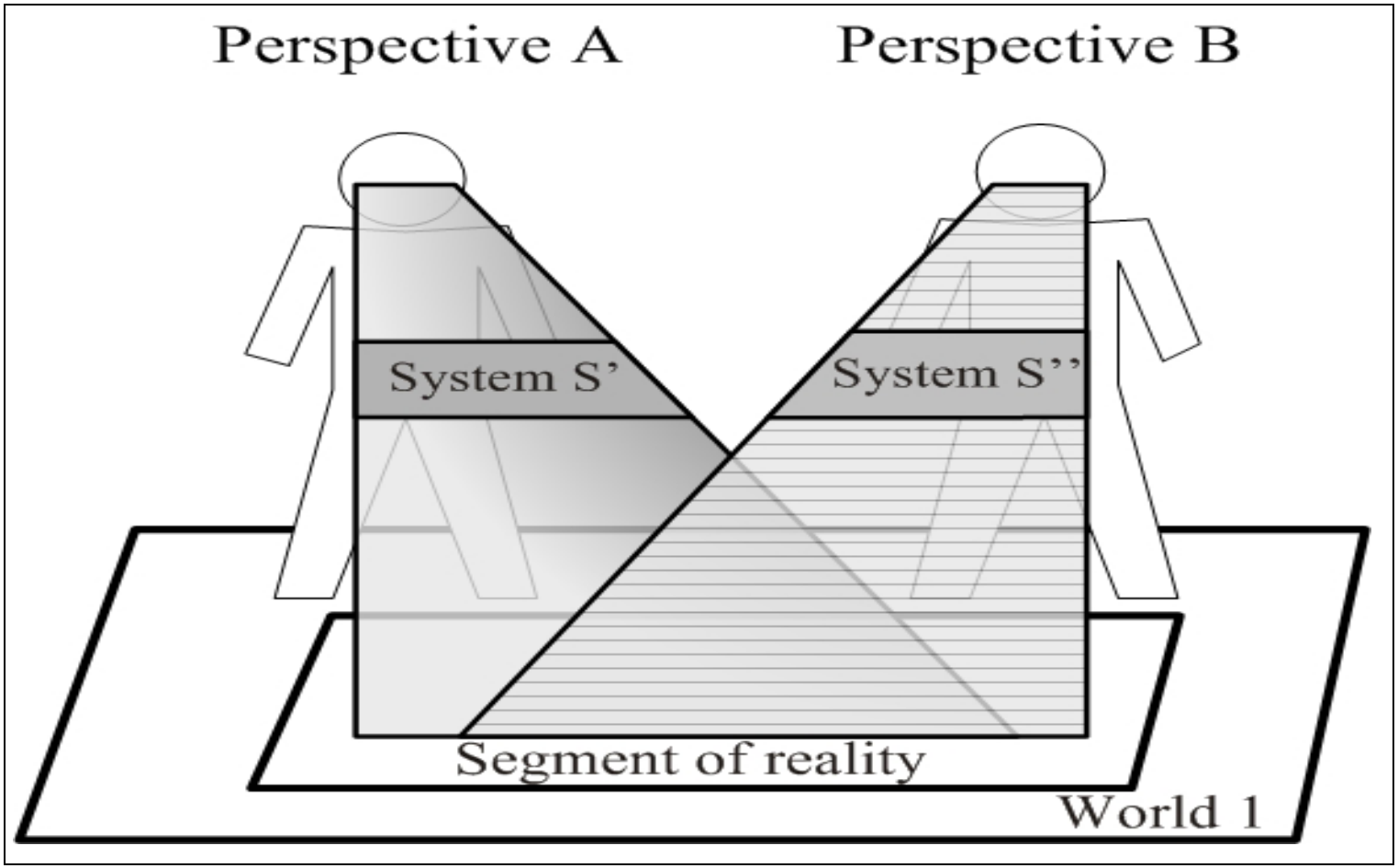
psycho-social,
→ mutual influence of observer and observandum (1.1)

dynamic
→ temporal dynamics of organizations (1.2)

organizational information systems.

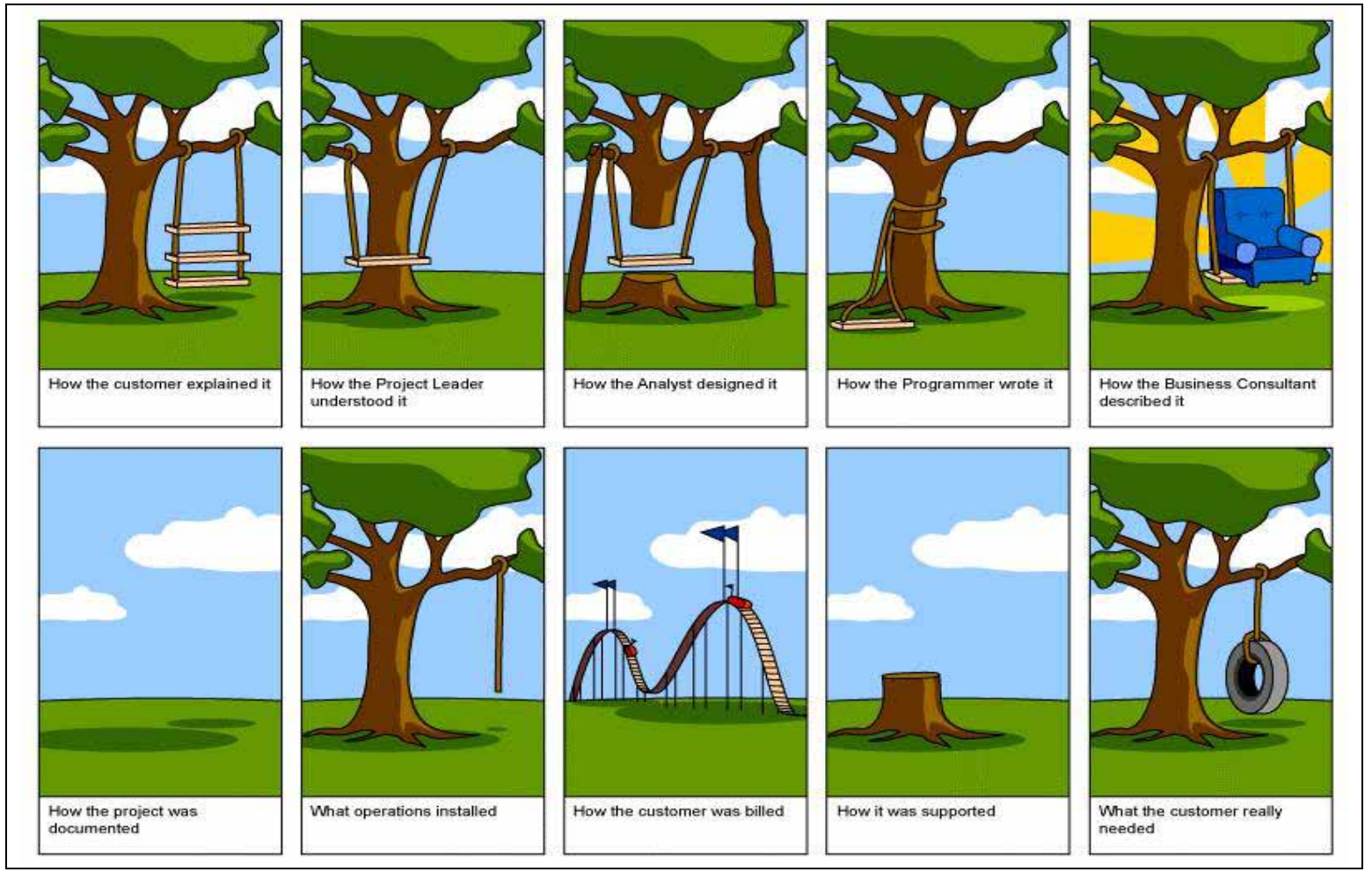
In this presentation, 'IS' always means the science IS or a technical IS.

1.1 Cooperative knowledge gain:
multi-perspectivity of IS experts and organization / domain experts 1



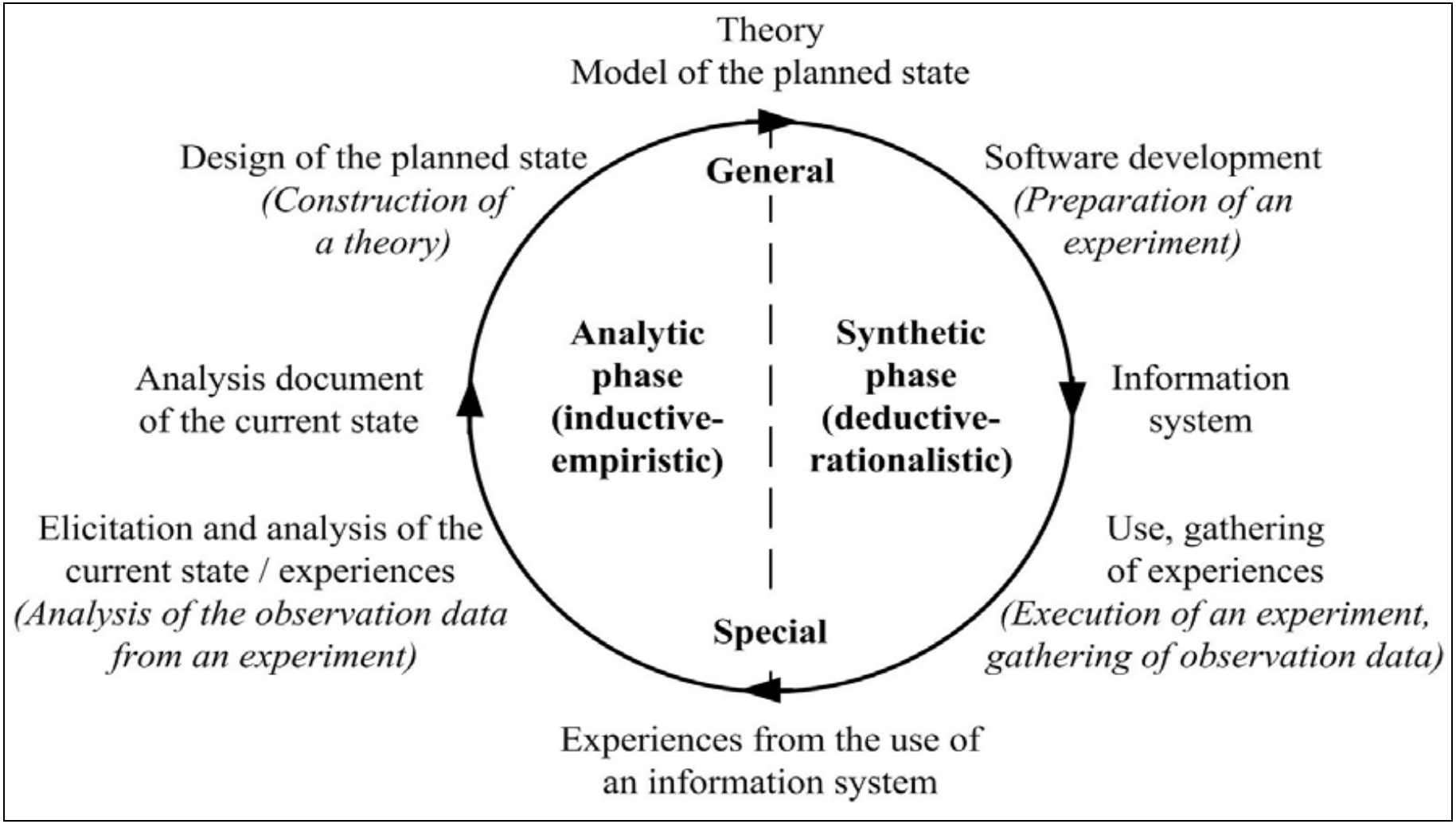
Systems are relative to perspectives (adapted from Steinmüller 1993, 168)

1.1 Cooperative knowledge gain: multi-perspectivity of IS experts and organization / domain experts 2



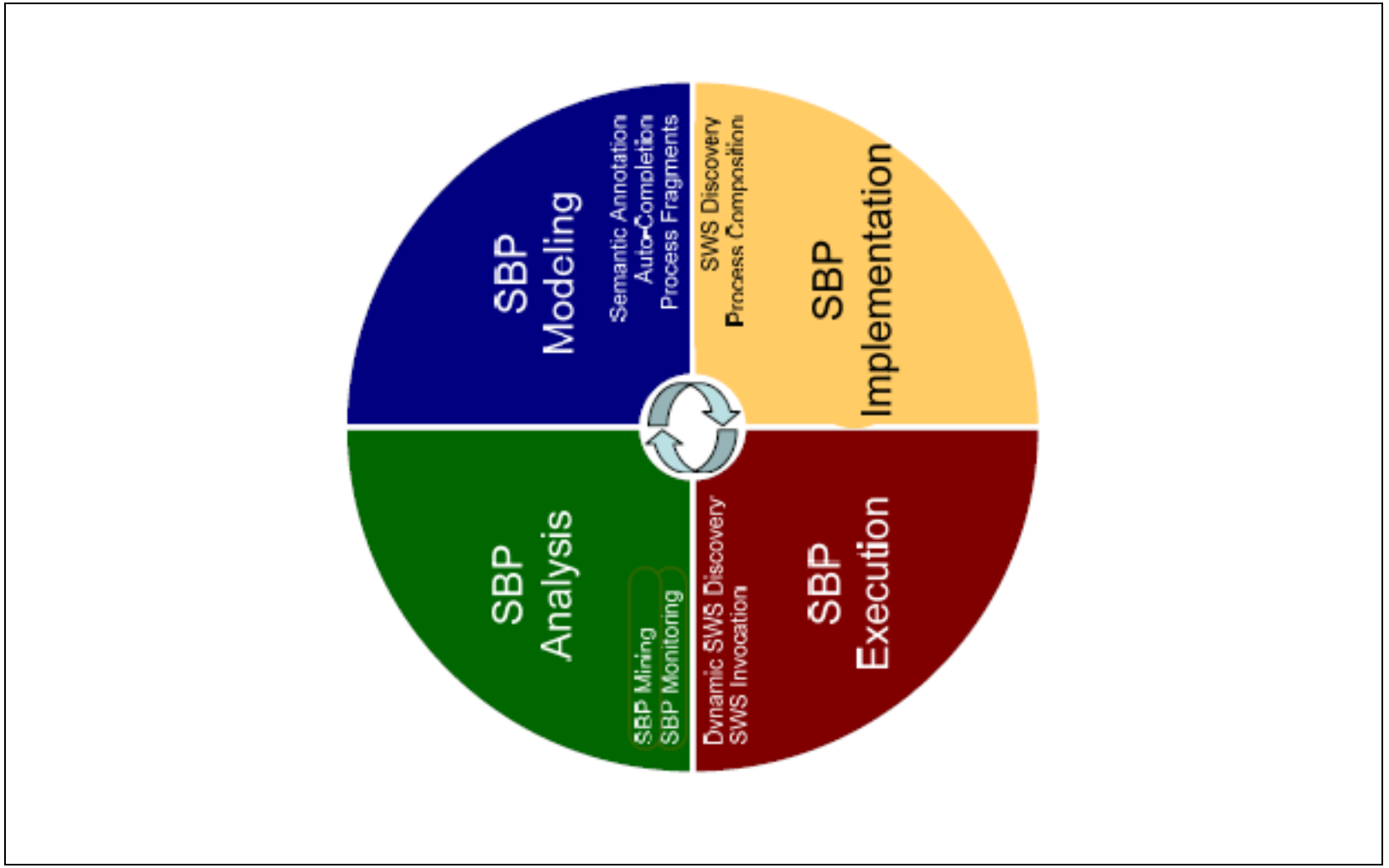
Multi-perspectivity in project management (passim on the Internet)

1.2 Cyclic knowledge gain: mayeutic cycle 1



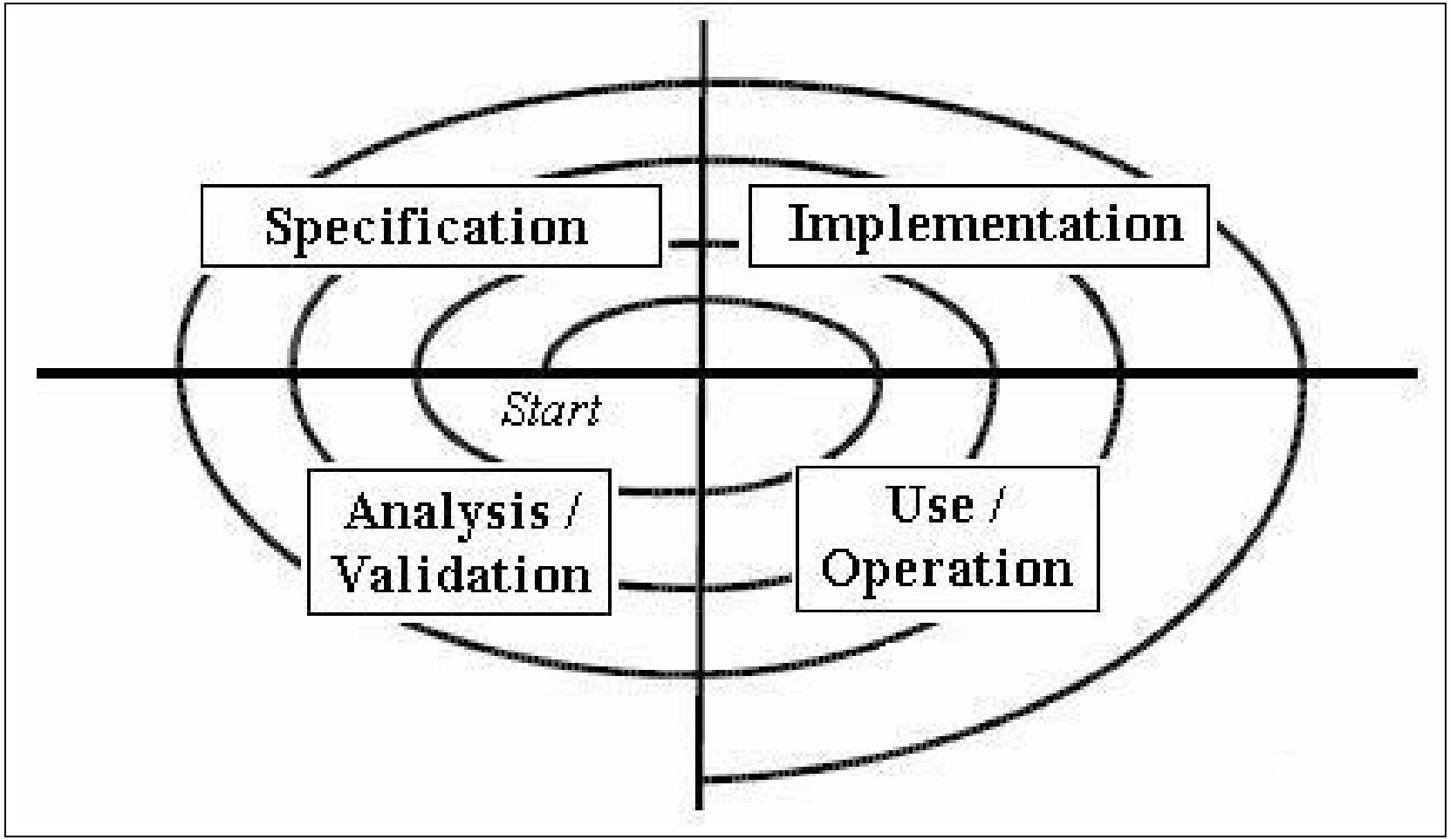
Mayeutic cycle in IS and natural sciences (according to Holl 1999, 175)

1.2 Cyclic knowledge gain: mayeutic cycle 2



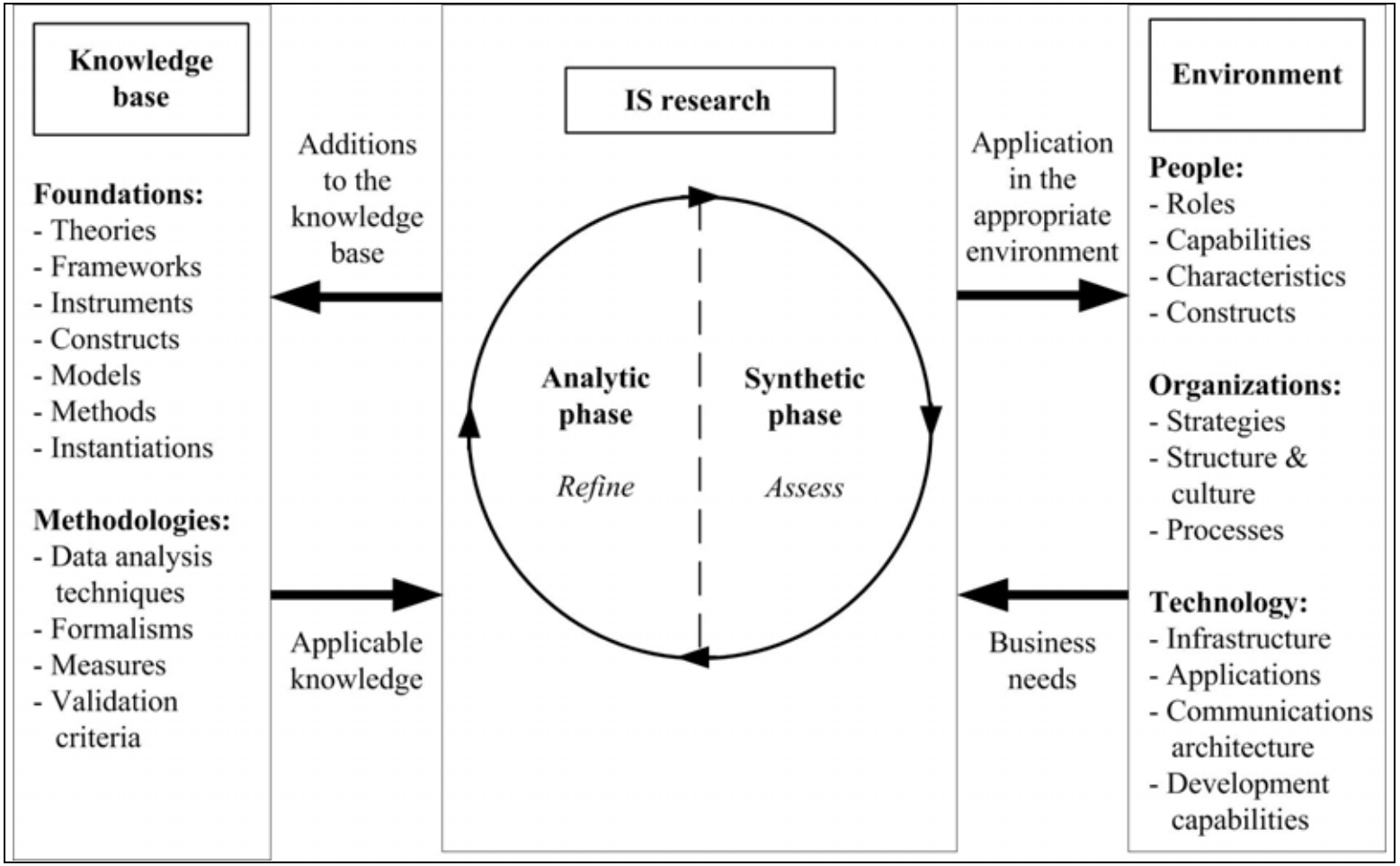
BPM life cycle (Wetzstein et al., SBPM 2007, p. 4)

1.2 Cyclic knowledge gain: mayeutic cycle 3



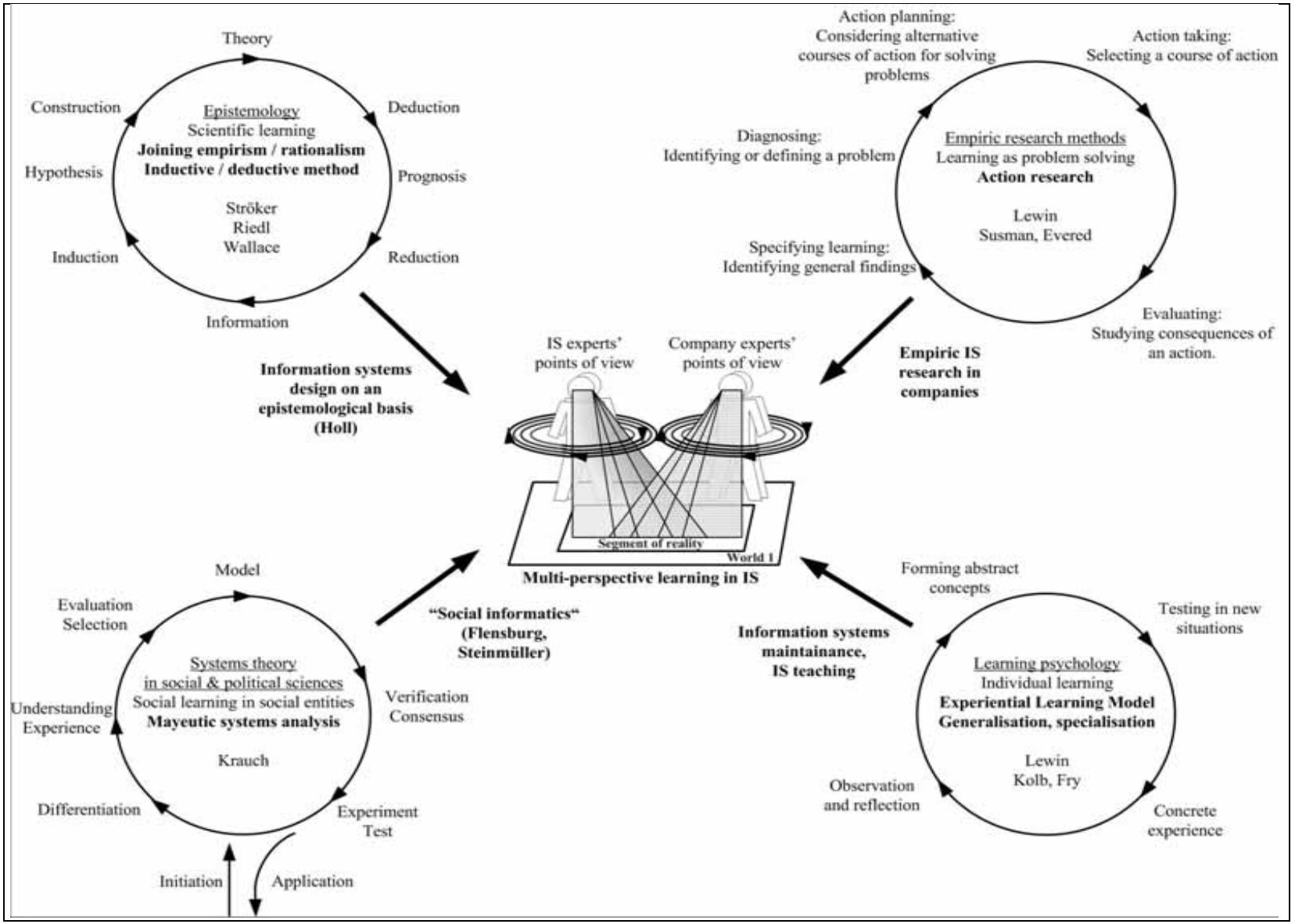
A spiral model of software development and enhancement
(adapted from Sommerville 2001, 614 according to Boehm 1988)

1.2 Cyclic knowledge gain: mayeutic cycle 4

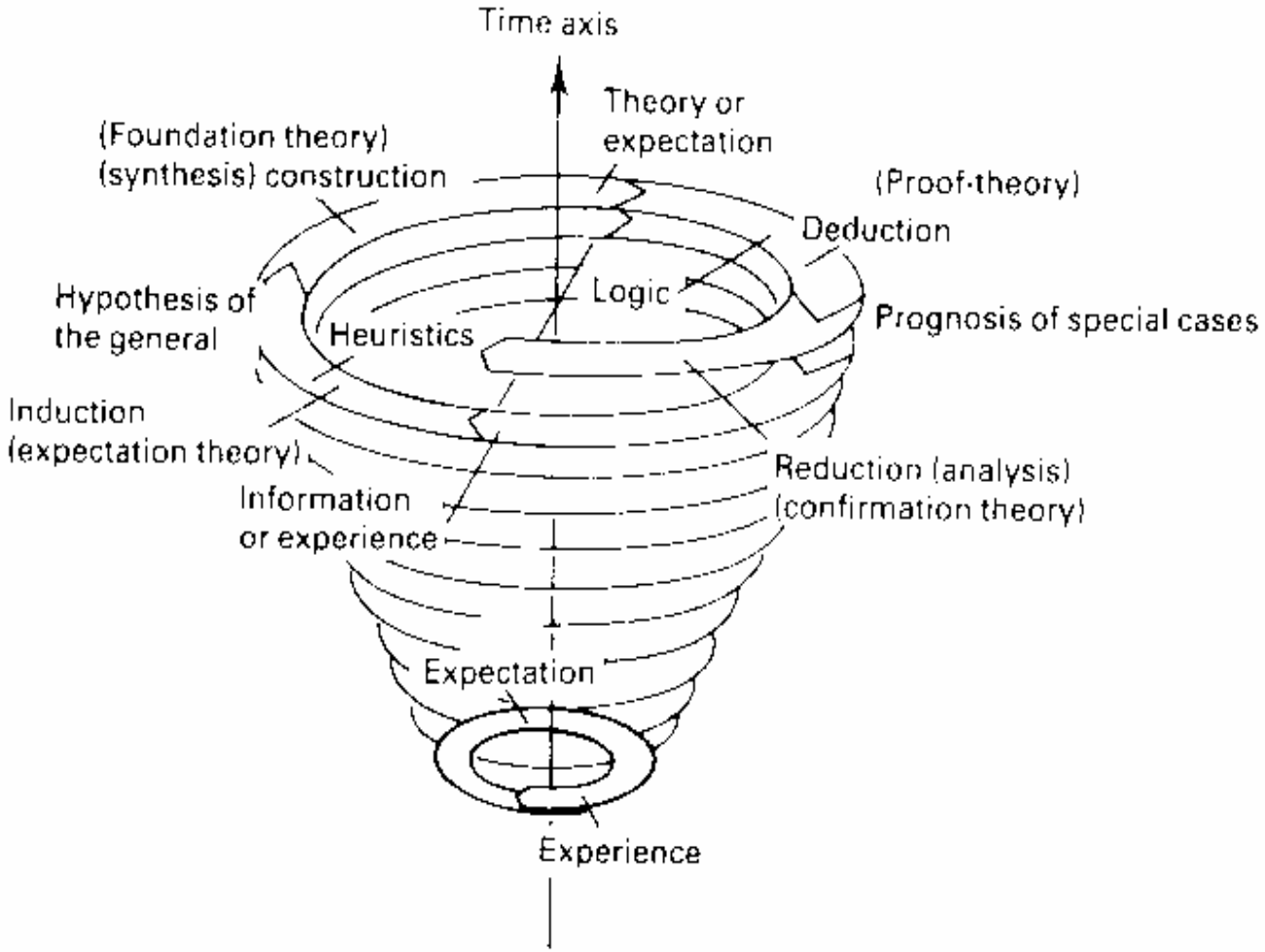


Embedded mayeutic cycle in IS research
(adapted from Hevner / March / Park / Ram 2004, 80)

1.2 Cyclic knowledge gain: mayeutic cycle 5: cycles of knowledge gain



1.2 Cyclic knowledge gain: mayeutic cycle 6



The cycle of cognitive gain, hence the growth of knowledge and certainty according to Erhard Oeser's theoretical system-functional model of the dynamics of theories. The symmetries contained in this algorithm correspond to those which were found prepared in the phylogeny of biological cognitive processes. It is only that they are more differentiated at the level of epistemology (from Oeser 1976; extended into biological history).

Conic helix of knowledge gain (Riedl, Biology of knowledge 1984, 169)

2. IS anti-aging supported by cooperative cyclic knowledge gain

2.1 Fundamental definitions

Permanent changes on the organizational level
require permanent changes on the IT level
which lead to software aging.

2.1.1 Types of software aging (Parnas 1994, 280)

- functional aging (lack of movement): no changes at all
- qualitative aging (ignorant surgery): important in this context

2.1.2 Lehman's SPE classification (Lehman 1980, 1061-1063)

In the area of IS,
only Lehman's E-type systems / programs are considered.
(→ 2.4.2 Lehman's laws of software evolution)

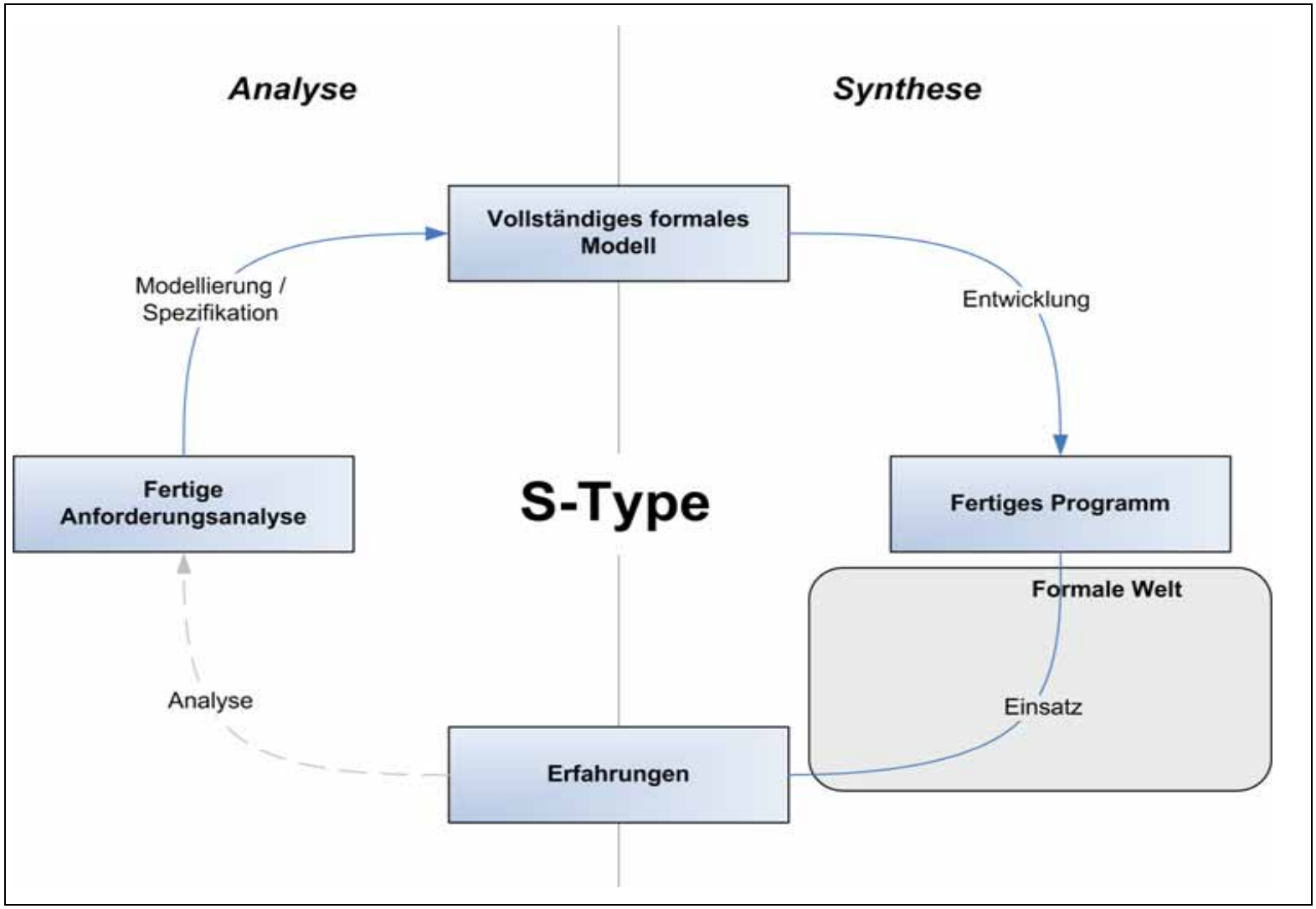
2.1.2.1 S-type systems: specifiable

An IT system belongs to type S if one can prove that a previous specification is mathematically correct.

- Requirements are described completely
- Problem does not change
- Acceptance: mathematical correctness
- Improvement is impossible

Examples: inversion of matrices; solution of equations, World-3 problems

2.1.2.1 S-type systems



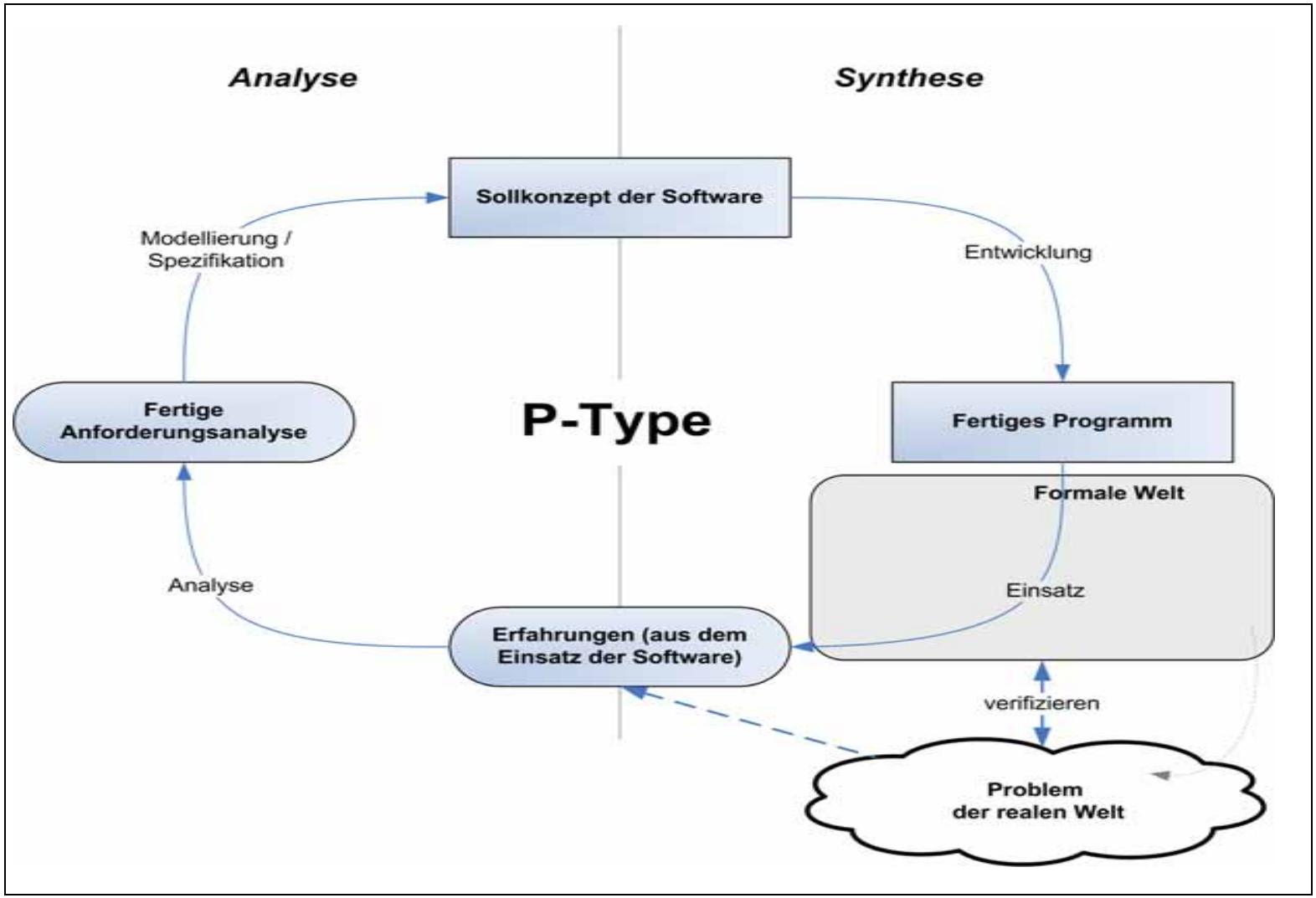
2.1.2.2 P-type systems: problem-solving

P-type systems are solutions for limited problems which cannot be described completely on a formal level.

- A complete formalization is impossible
- The problem is simplified
- The problem on reality level is not solved
- Acceptance via use
- Continuous improvement

Examples: weather forecast; World-1 problems

2.1.2.2 P-type systems



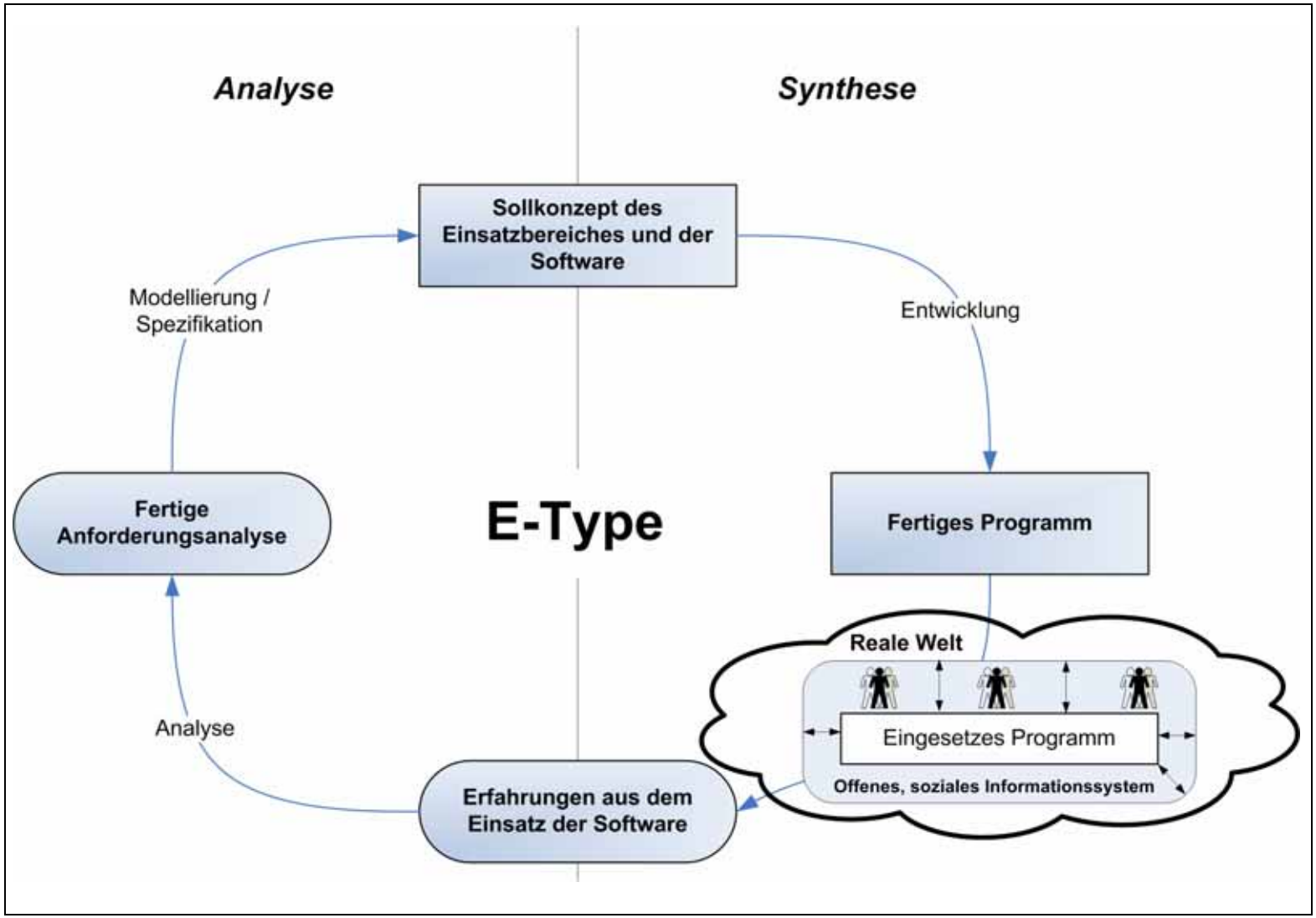
2.1.2.3 E-type systems: embedded

E-type systems are embedded in open, dynamic, complex, social (socio-technical) information systems (organizations).

- Automation of human or social activities
- Requirements are not clear
- Acceptance: the user is content
- Continuous improvement

Examples: business information systems; World-2 problems

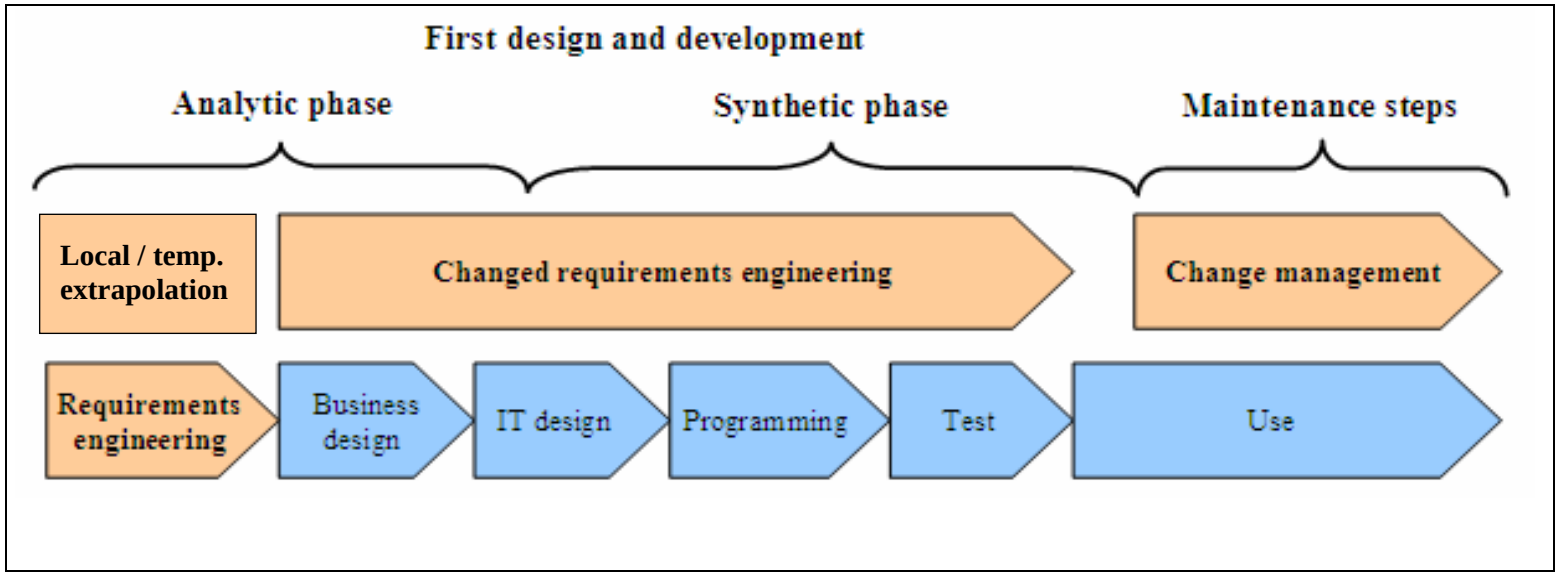
2.1.2.3 E-type systems



2.1 Fundamental definitions

Every anti-aging method mentioned in the following sections
will cover each of the two knowledge gain aspects from Chapter 1:
- cooperative
- cyclic

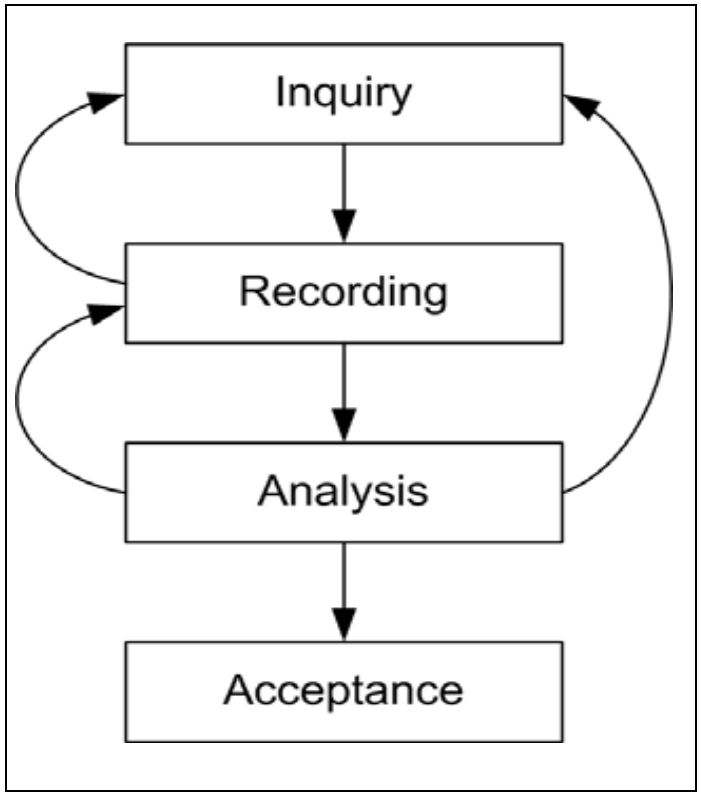
Overview



Requirements and IS life cycle (Holl / Paetzold / Breun 2011, p. 33)

2.2 IS anti-aging methods during the analytic phase 1

2.2.1 Requirements engineering 1



Life cycle of a requirement
(Holl / Maydt 2007, 6; adapted from Partsch 1998, 27)

2.2 IS anti-aging methods during the analytic phase 2

2.2.1 Requirements engineering 2

The psychiatric approach of neuro-linguistic programming
by R. Bandler / J. Grinder used in Rupp, Chris: RE. München 1st ed. 2001
tries to find the
underlying meaning of utterances produced by means of transformations
(cf. Freud's projection to others and exaggerated contrary)

Deletion

Implicit assumptions, incomplete properties:
easily changeable: how easily? what's easy? by whom?
Incomplete verbal nuclei/kernels
(missing objects and adverbials; cf. dependence grammar):
SW shall inform about errors: whom? where? how? when?
the development of a SW tool: who develops? when? why?

2.2 IS anti-aging methods during the analytic phase 3

2.2.1 Requirements engineering 3

Generalization

Universal quantifiers:

every error: really every? any exceptions?

Incomplete conditions:

If the error X occurs in the last phase of program Y, then ...

What should be done if it occurs in another phase?

Definite article without text reference:

the error: which?

Distortion

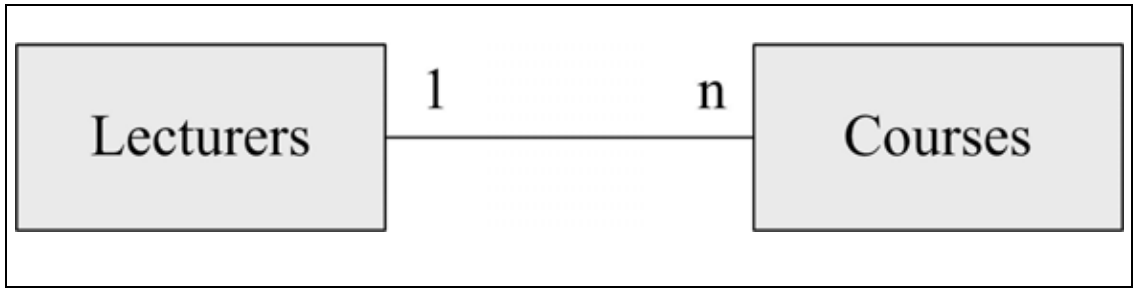
Nominalization: resulting event instead of process

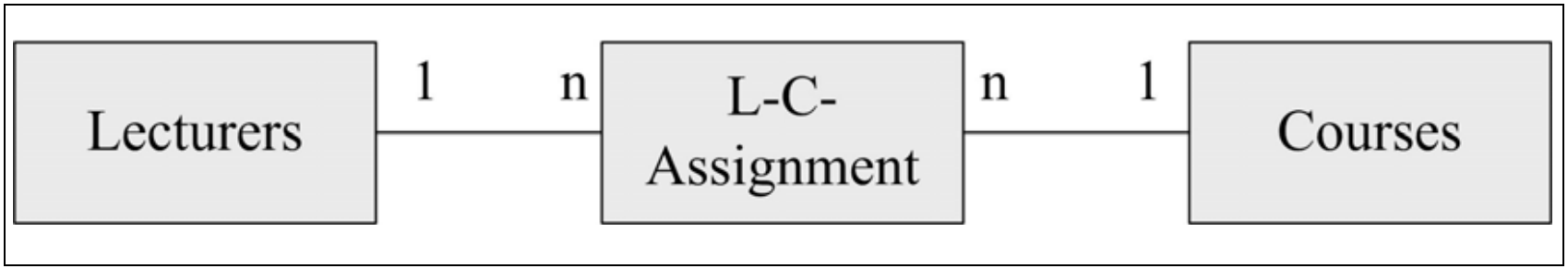
loss of data: which data are lost? when? how? why?

2.2 IS anti-aging methods during the analytic phase 5

2.2.2 Open and dynamic models 1: local extrapolation

Examination of other application areas which already exist



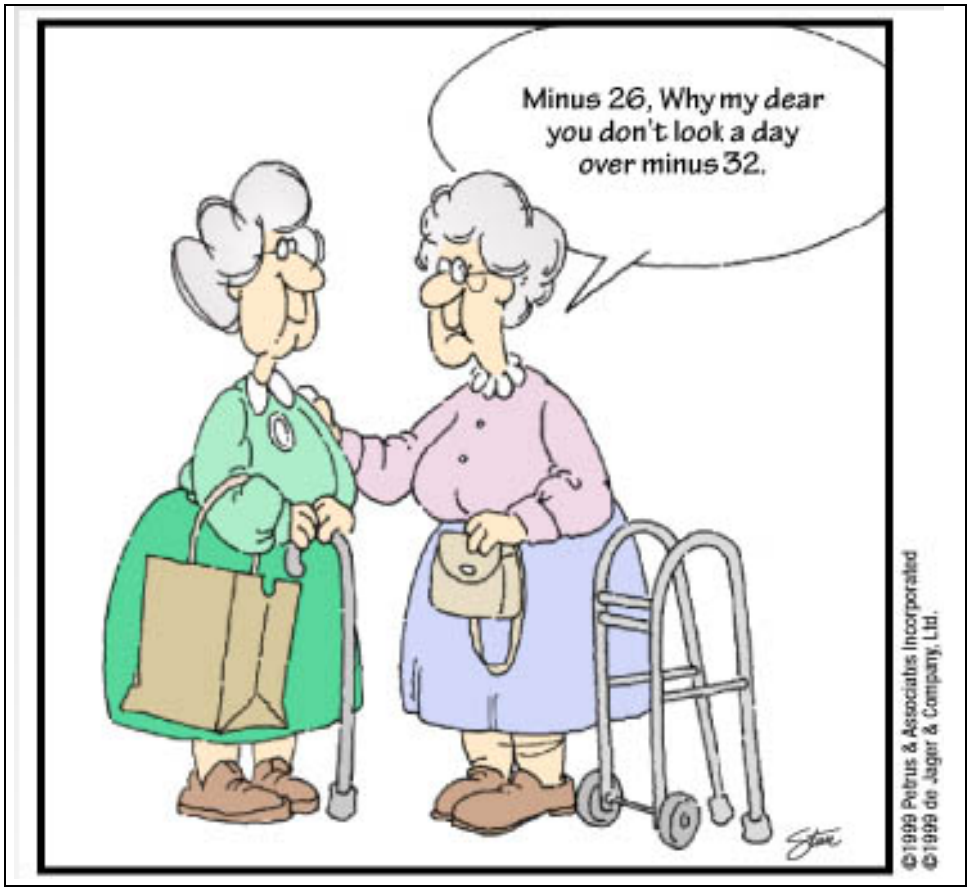


Data model before and after local extrapolation
(Holl / Paetzold / Breun 2011, p. 36-37)

2.2 IS anti-aging methods during the analytic phase 6

2.2.2 Open and dynamic models 2: temporal extrapolation

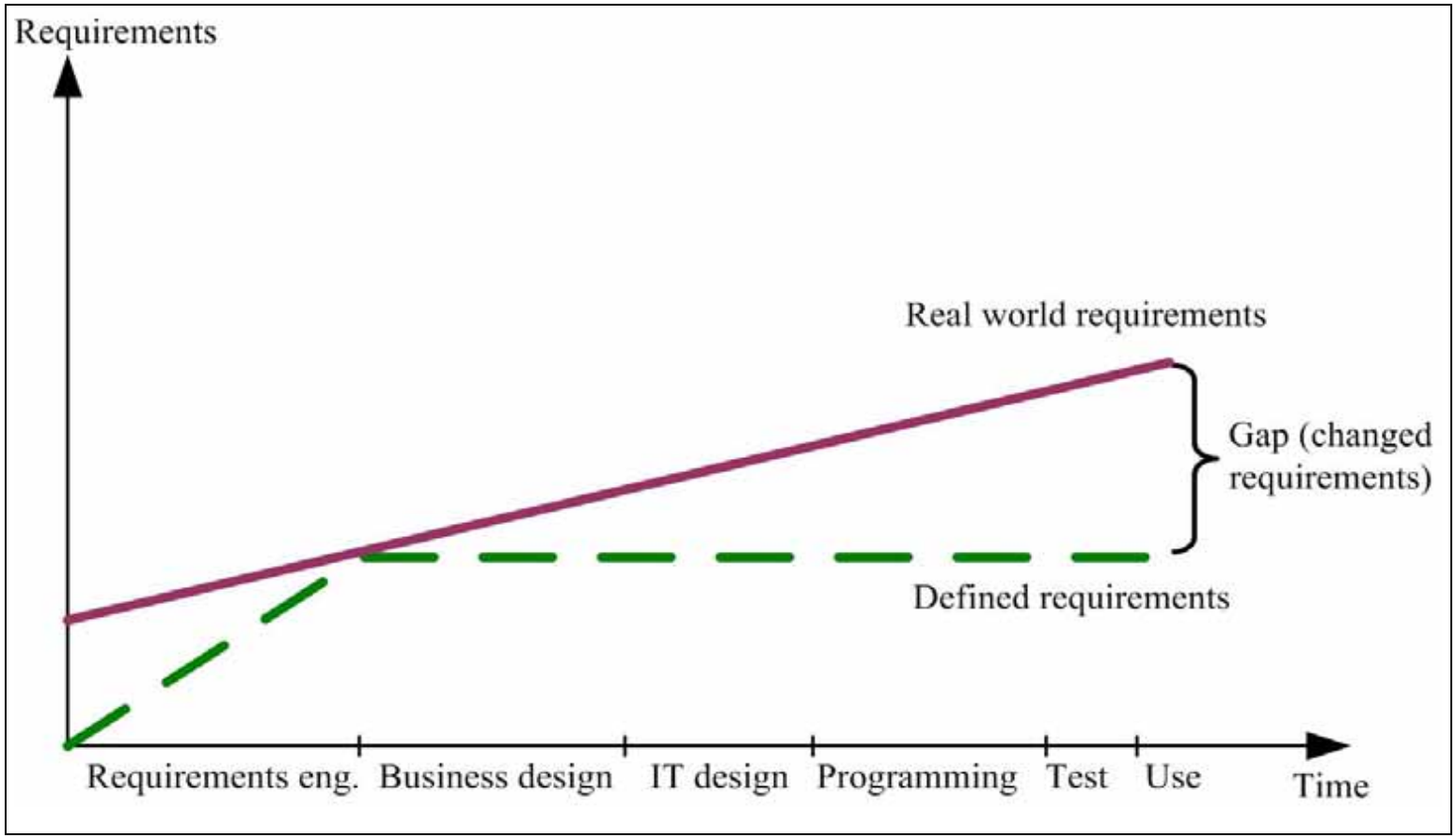
Examination of future developments



Millennium bug (Y2K)

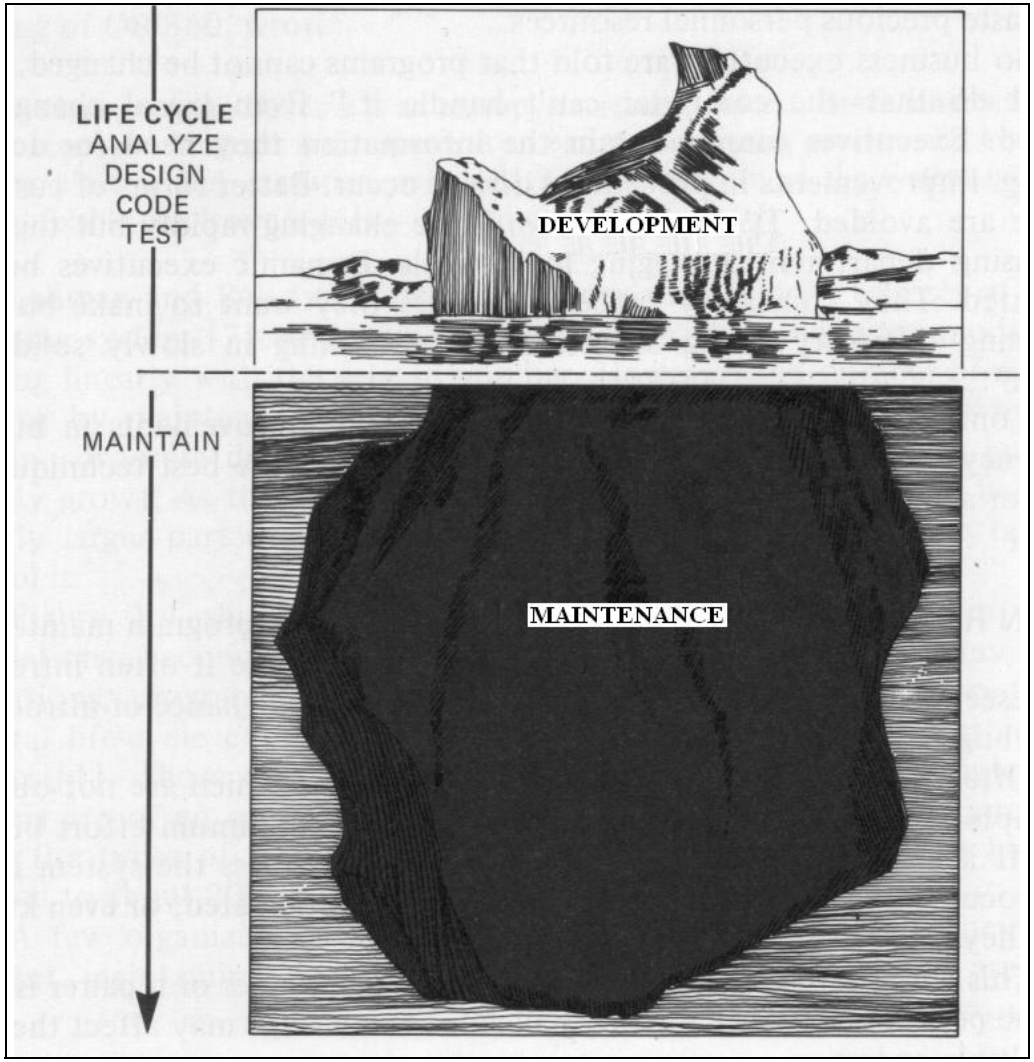
2.3 IS anti-aging methods during the synthetic phase

Changed / creeping requirements management / engineering



Changed requirements gap (Holl / Paetzold / Breun 2011, p. 39)

2.4 IS anti-aging methods during maintenance



The iceberg of IS maintenance (Martin / McClure, 1983, p. 7)

2.4 IS anti-aging methods during maintenance

2.4.1 Types of IS maintenance

(refined from Swanson 1976, 492-497 and Kroha 1997, 181)

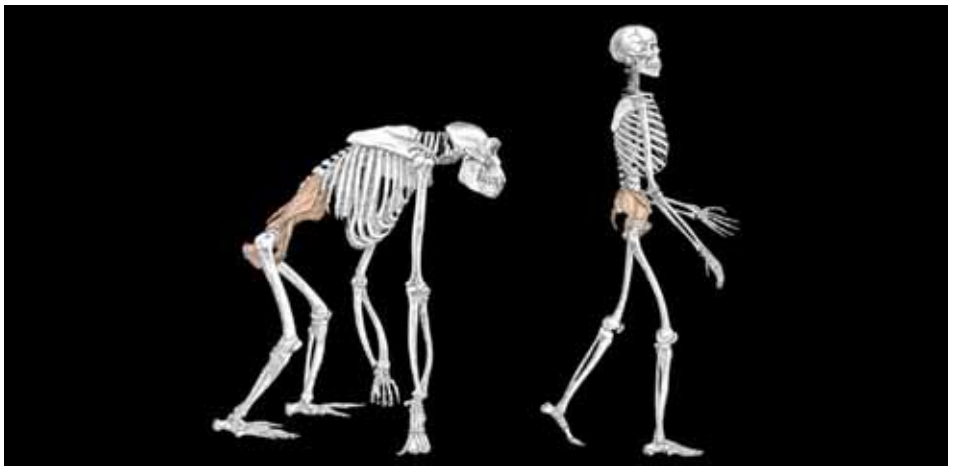
Only types specific for E-type systems are considered.

technical-corrective
business-corrective
technical-adaptive
business-adaptive → change management
technical-perfective → reengineering
business-perfective → vs. excellent requirements engineering
technical-preventive
business-preventive → extrapolation

2.4.2 IS maintenance types and their relation to Lehman’s laws

2.4.2.1 Software Evolution

Change of IT systems during their life cycle
vs. biologic evolution: change of species
Ontogenesis vs. phylogenesis

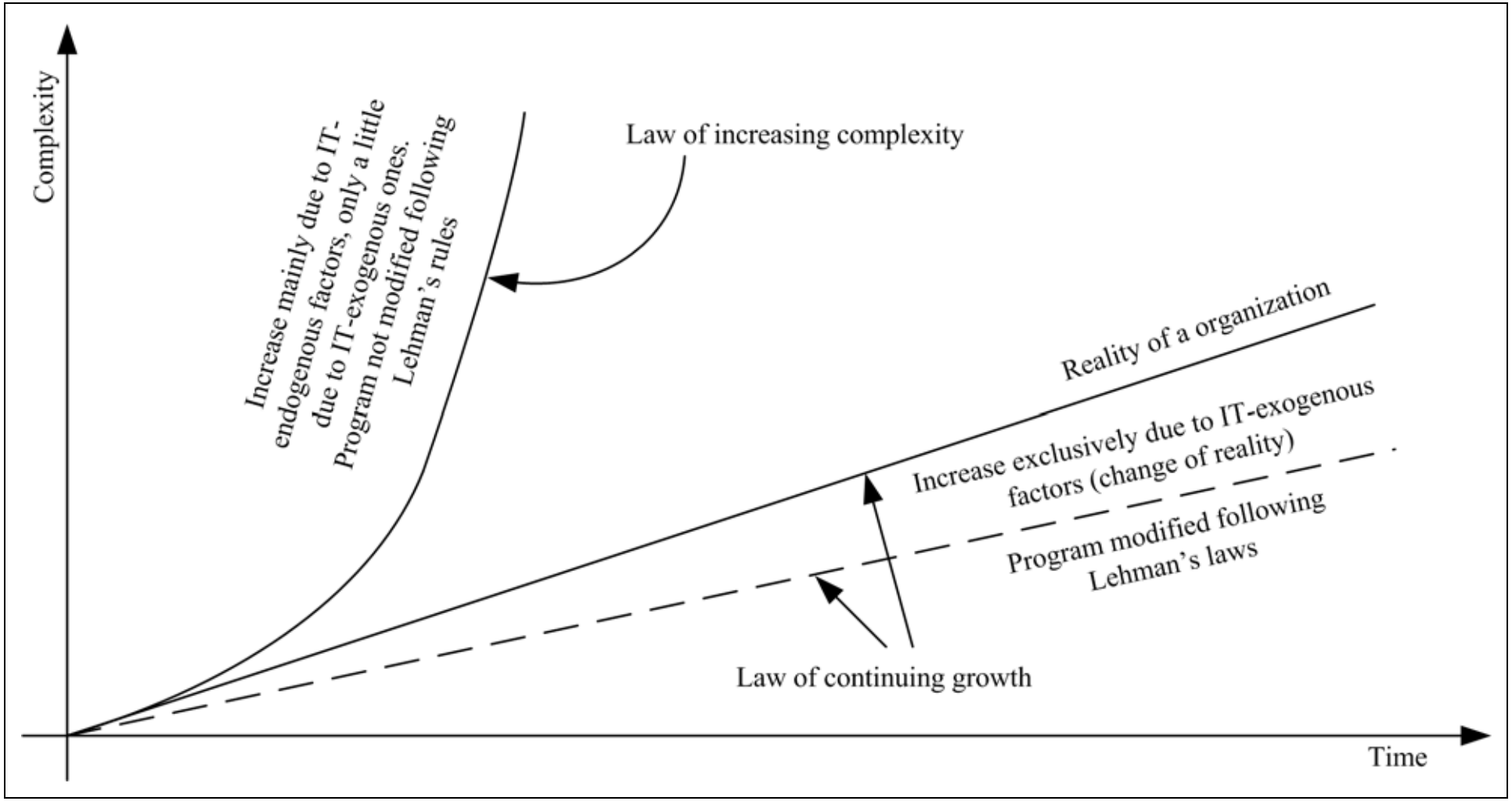


2.4.2.2 Lehman's laws of software evolution

Law	Description	Year
I	Continuing change	1974
II	Increasing complexity	1974
III	Self regulation	1974
IV	Conservation of organizational stability	1980
V	Conservation of familiarity	1980
VI	Continuing growth	1980
VII	Declining quality	1996
VIII	Feedback system	1996

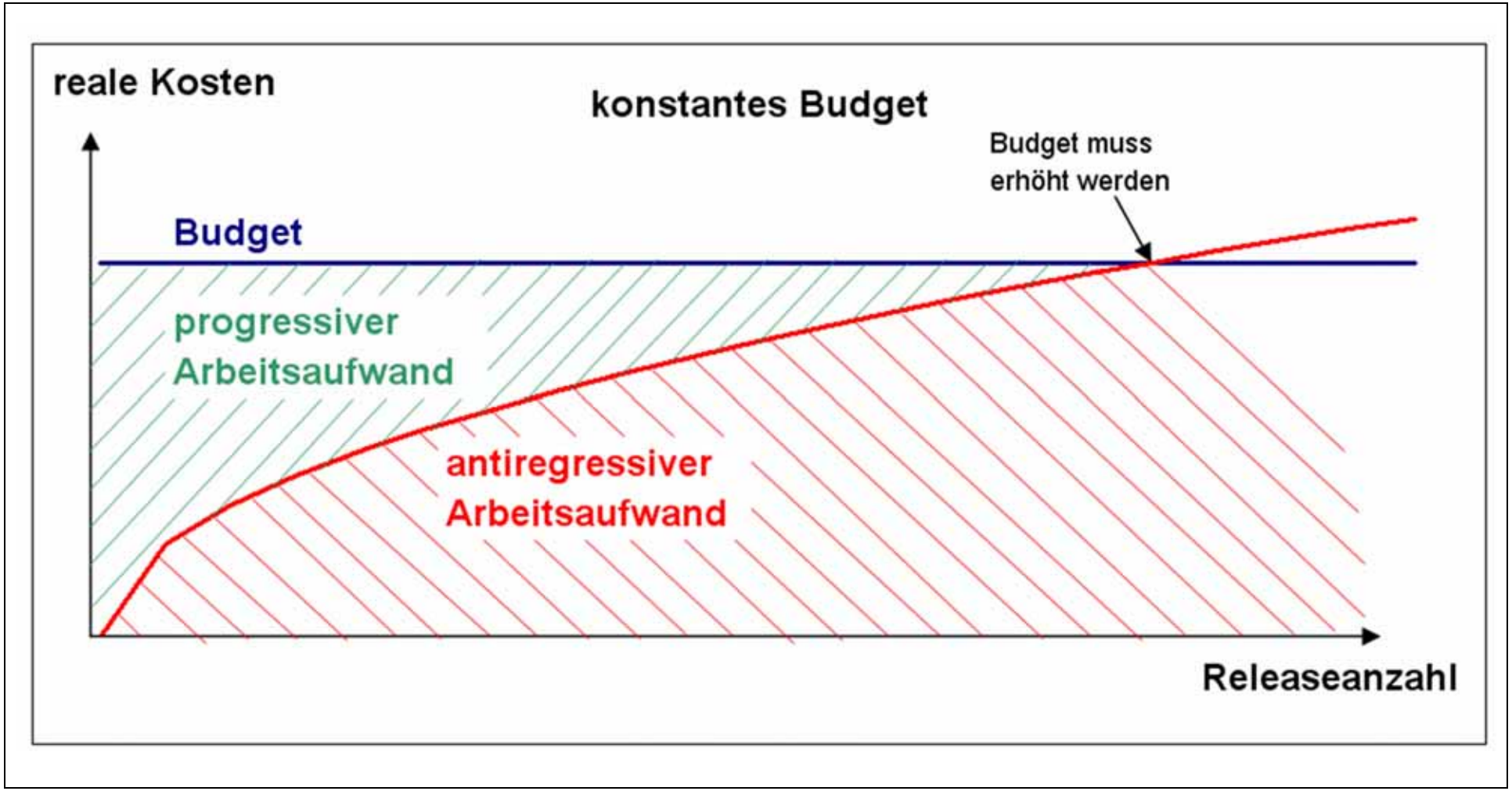
(Lehman / Belady 1972 etc.)

2.4.2.3 Applications of Lehman's laws 1 (legacy systems)



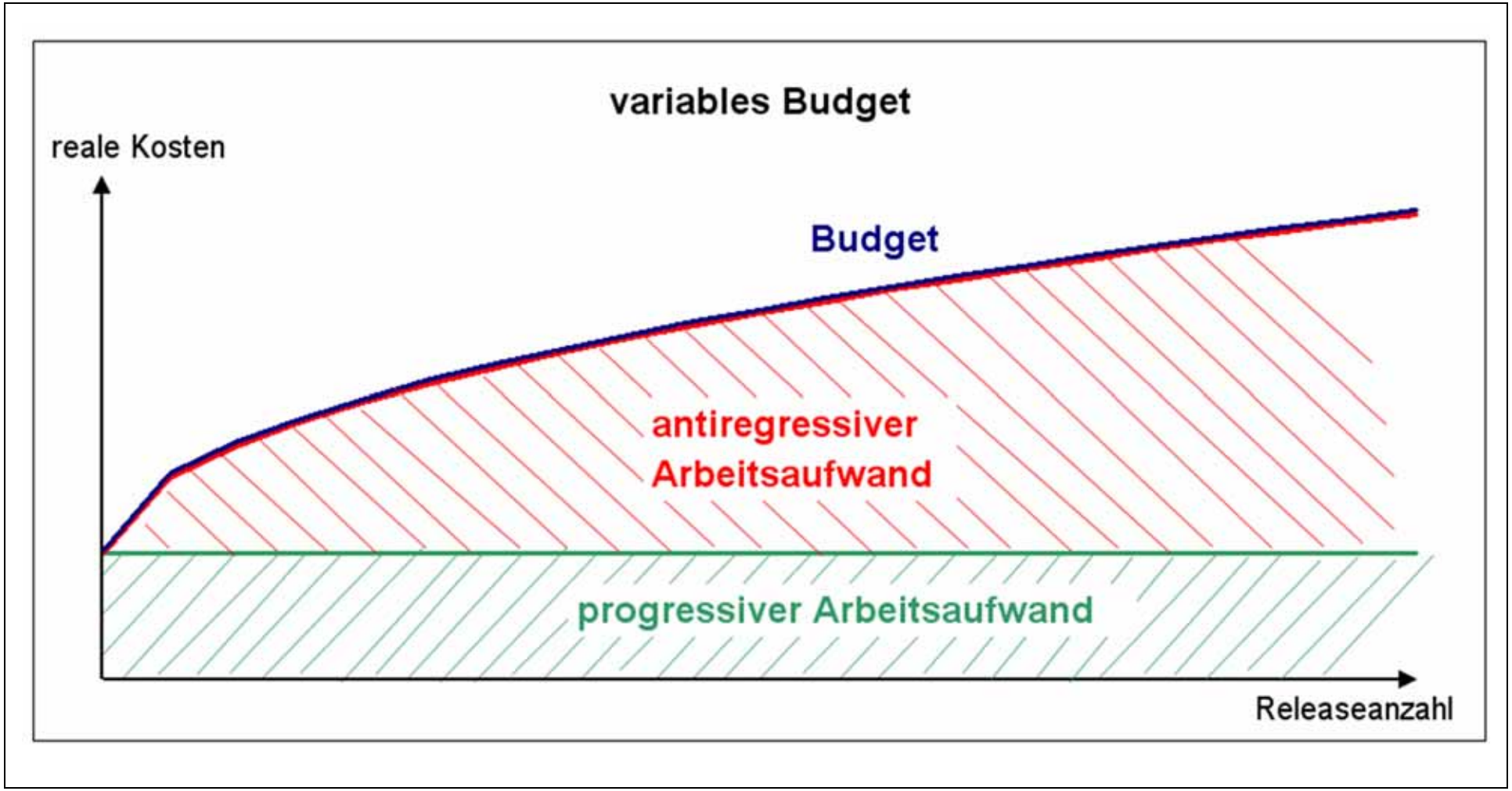
Increasing complexity of E-type systems (Holl / Paetzold / Breun 2011)

2.4.2.3 Applications of Lehman's laws 2



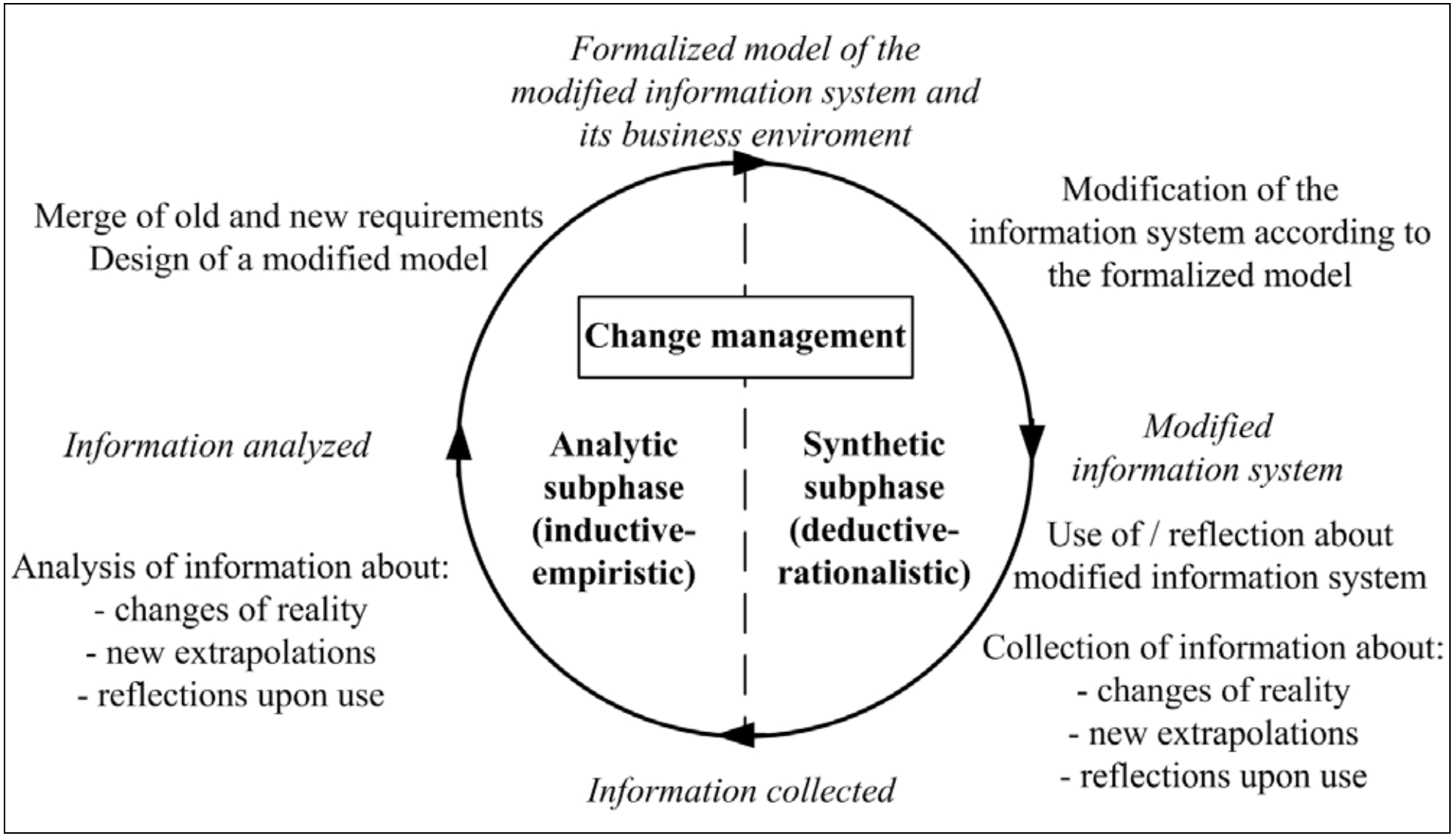
Progressive and anti-regressive costs over time

2.4.2.3 Applications of Lehman's laws 3



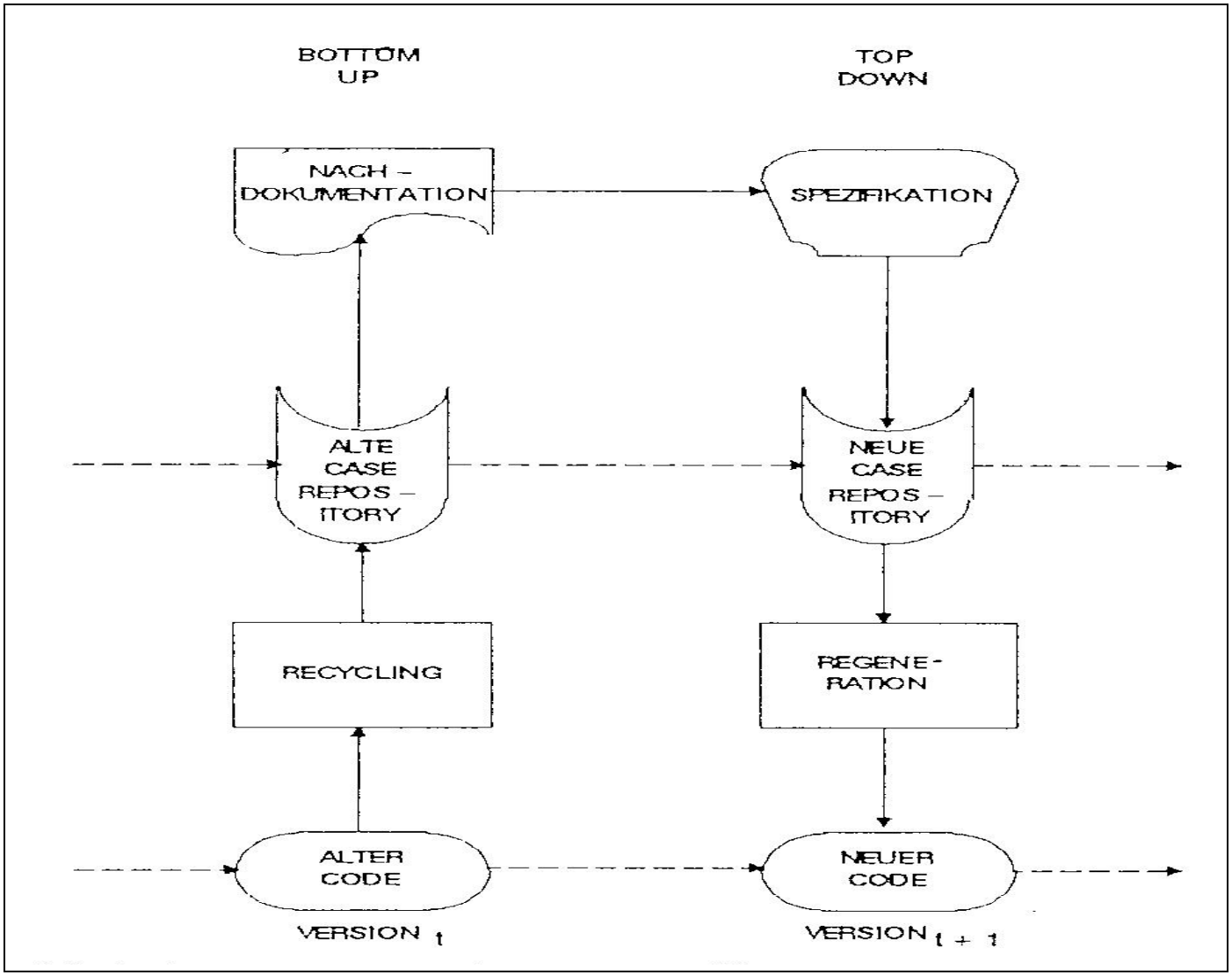
Progressive and anti-regressive costs over time

2.4.3 Change management



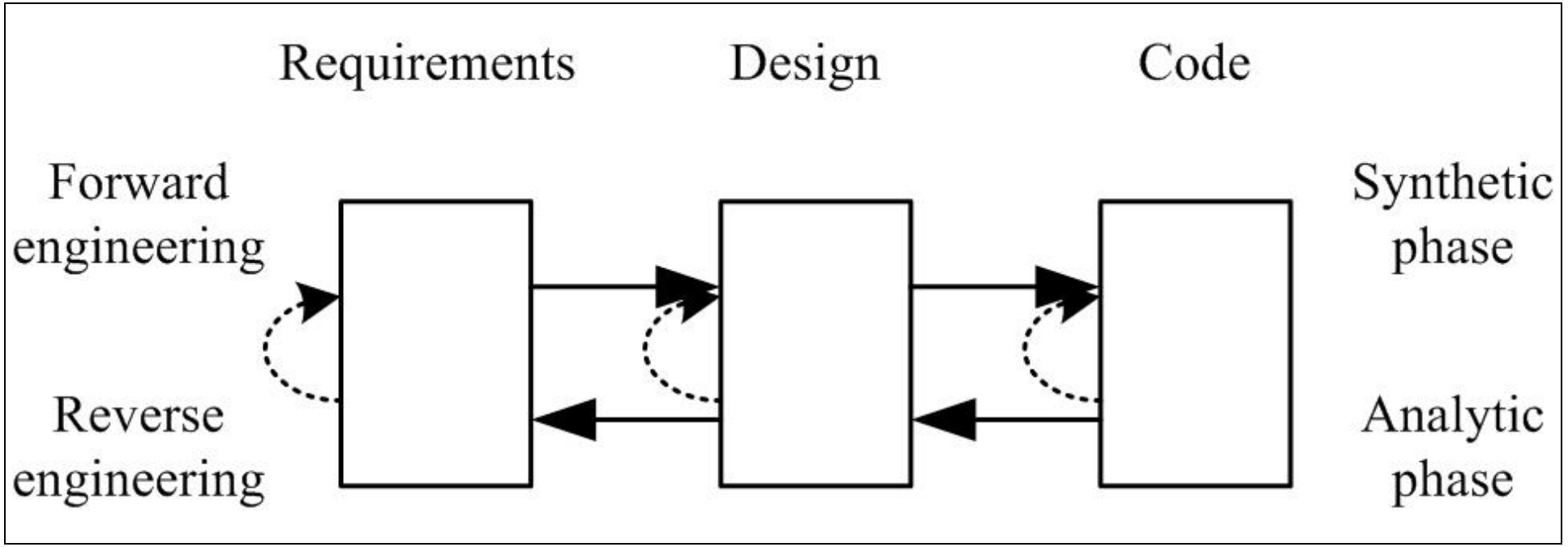
Mayeutic cycle of IS maintenance (Holl /Paetzold / Breun 2011, p. 45)

2.4.3 Change management



Bottom-up SW development (Sneed, SW maintenance, 1991, fig. 2.10)

2.4.4 Reengineering



Reengineering process
(adapted from http://mlecture.uni-bremen.de/intern/ws2005_2006/fb03/vak-03-706.1/20051024/folien.pdf
according to Chikofsky / Cross II, 1990)

2.5 Summary

Anti-Aging starts
with the beginning of the analytic phase of the software process
and has to be considered
throughout the entire software process and every maintenance step.

3. References

Boehm, B. W. (1988). A spiral model of software development and enhancement, IEEE Computer, 21(5), 61-72.

Chikofsky, E. J. and Cross II, J. H. (1990). Reverse engineering and design recovery. A taxonomy. IEEE Software, 7(1), 13-17.

Hevner, A. R. , March, S. T., Park, J. and Ram, S. (2004). Design science in information systems research. MIS Quarterly, 28(1), 75-105.

Holl, A.; Paetzold, F.; Breun, R. (2011). Cooperative cyclic knowledge gain in IS anti-aging. Nuremberg: University of Applied Sciences.

Holl, A.; Maydt, D. (2007). Epistemology in information systems: Modeling and requirements engineering. In A. Erkollar (Ed.), State of the art: Enterprise and business management. A handbook for educators, consulters and practitioners. Marburg: Tectum.

Holl, A. (1999). Empirische Wirtschaftsinformatik und evolutionäre Erkenntnistheorie. In J. Becker et al. (Eds.), Wirtschaftsinformatik und Wissenschaftstheorie. Bestandsaufnahme und Perspektiven (pp. 163-207). Wiesbaden: Gabler.

Kroha, P. (1997). Softwaretechnologie. München: Prentice Hall.

Lehman, M. M. (1980). Programs, life cycles and laws of software evolution. Proceedings of the IEEE, 68(9), 1060-1076.

Lehman, M. M. and Belady, L. (1972). An introduction to program growth dynamics, statistical computer performance evaluation. In W. Freiburger (Ed.), Statistical computer performance evaluation (pp. 503-511). New York: Academic Press.

Martin, J.; McClure, C. (1983). Software maintenance: The problem and its solution. Englewood Cliffs, New Jersey: Prentice-Hall.

Parnas, D. L. (1994). Software aging. Proceedings of the 16th International Conference on Software Engineering (ICSE) (pp. 279-287). Los Alamitos, CA: IEEE Computer Society Press.

Partsch, H. (1998). Requirements Engineering systematisch – Modellierung für softwaregestützte Systeme. Berlin: Springer.

Riedl, R. (1984). Biology of knowledge. The evolutionary basis of reason. Chichester: Wiley.

Sommerville, I. (2001). Software engineering (6th ed.). München: Pearson Studium.

Steinmüller, W. (1993). Informationstechnologie und Gesellschaft: Einführung in die angewandte Informatik. Darmstadt: Wissenschaftliche Buchgesellschaft.

Swanson, E. B. (1976). The dimensions of maintenance. Proceedings of the IEEE/ACM Second International Conference on Software Engineering (pp. 492-497).