Alfred Holl

Principles of IS modeling

1 Motivation 1

1.1 Relation between IT and organizations

Information systems (IS)
– as a science and
– as technical systems
have the task
to systematically optimize information processing
in organizations (non-profit and profit, that is, enterprises),
to support business tasks in organizations.

Organizations, however, are open, dynamic, complex, socio(-technical) information(-processing) systems.

Therefore, business tasks in organizations are not so formal that a complete support by IT would be possible:
The organization level is always broader than the IT level.

organization level (social IS)
IT level (technical IS)

Technical IS do not possess any technological end in itself.
The purpose is not to just introduce technology into business environments.
The organization level has a clear priority:
Business IT alignment, IT Governance

There is no need that technology is part of a technical IS:
There are technical IS without IT, e.g. card indexes.

1 Motivation 2

Therefore, we need methods (1.2) to systematically
– produce technical information systems
– design models (systems analysis, requirements engineering, business process modeling, reference modeling, OO analysis)

1.2 Methods in IS

Methods to professionally, systematically and economically produce technical information systems
always include the
– support / management aspect (project management) and
– engineering / scientific / technical aspect (software engineering).

Project management has the task
to manage
– personnel teams, human resources psychology
– time and money
– documentations.

Software engineering has the task
to professionally, systematically and economically produce software systems, such as technical information systems.

In this sense, software engineering means
a lot more than software technology and
a lot more than applying graphic symbol systems, representation syntaxes, such as UML.

1 Motivation 3

1.2 Methods in IS

Methods to professionally, systematically and economically design models are closely related to epistemology.
They are not sufficiently considered of today's software engineering.

Level	Partly methodic, partly structured	Epistemology-based	Epistemological foundation
Eliciting the current state	Systems analysis, Reverse engineering	(Missing)	Systems theory
Designing the planned state	Business concept modeling	Requirements engineering	Linguistics, psychology, …

(Holl / Maydt, Epistemological foundations of RE, 2007, 48)

2 How to systematically design a technical IS 1

Basically, an information system is some (open) information-processing system.
It can be human / social, technical or socio-technical.

2.1 Principle of key and lock: compatibility of IT and application field 1

We cannot design isolated technical information systems.
They have to be embedded in some organizational environment.

Opposition:
Techn. IS are formal tools and fit only formal application areas.
Organizations are open, dynamic, complex, soci(o-technic)al information-processing systems
which comprise many informal parts (e.g. humans).

It is obvious
that tools have to be adapted to their application areas.
Formal tools, however, cannot be applied
in completely informal application areas.
Therefore, up to some extent,
the organization has to be adapted to the technical IS,
some formalization of the applying organization is inevitable.

The deployment of a technical IS
requires formalization of the application area.
Formalize (straighten) lock before modeling a formal key:
Business process reengineering

A mutual adaptation is necessary.

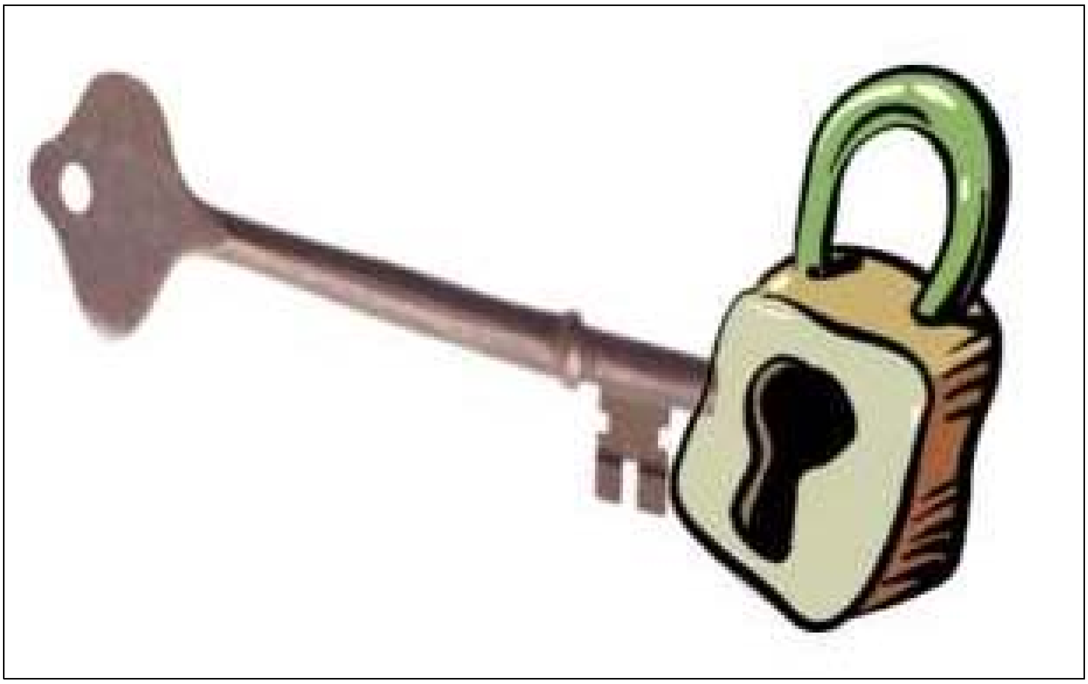
2 How to systematically design a technical IS 2

2.1 Principle of key and lock: compatibility of IT and application field 2

A good technical IS and its application area fit like key and lock in order to produce an efficient socio-technical system:

lock: organization level (social IS)
key: IT level (technical IS)

IT cannot cure the disastrous management of an organization.
A straight key cannot be put into a crooked lock.



Aim in 2.2-2.5:
an optimal technical information system
for a certain organizational environment (social IS).

2 How to systematically design a technical IS 3

2.2 Prescriptive model 1

Prescriptive models design a planned state to be achieved.

Before we can program a technical information system,
we have to design a technical information system.

Before we can design a technical information system,
we have to design its application area (organization).

A mere model of a technical information system is not sufficient.
As a technical IS is closely interrelated with its application area,
the application area itself has to be modeled as well.

The (conceptual) model of the planned state consists of:
– formal model of the lock (application area, organization:
at least social system, also socio-technical system)
– formal model of the key (technical information system)

Develop a clear idea of the technical IS's purpose and objectives.

The design of prescriptive models has to be done
in close cooperation with the domain experts involved
(participative strategies).

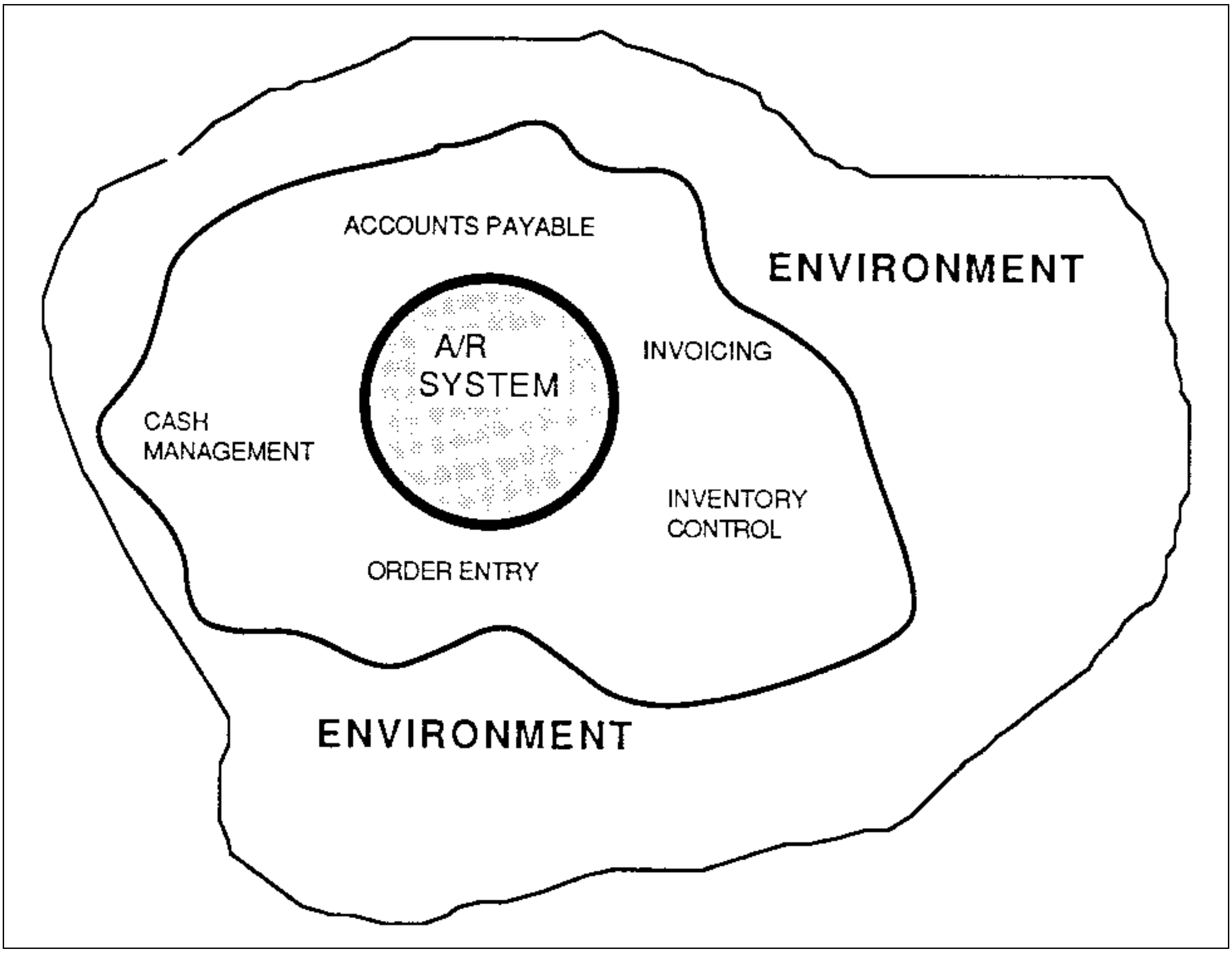
A good IS expert understands the application areas
of the technical information systems developed by himself so well
that he is able to use them in their application areas.

Methods for the design of the concept of the planned state:
requirements engineering on the basis of epistemology

2 How to systematically design a technical IS 4

2.2 Prescriptive model 2

**As organizations / departments are open systems,
always use a magnifying glass model (problem of isolation):
soft, blending system boundary
with precision / magnification decreasing towards the rim.**



**Accounts receivable system embedded in an enterprise
(Yourdon, Modern structured analysis, 1989, 336)**

2 How to systematically design a technical IS 5

2.3 Descriptive model

Descriptive models describe an existing current state.

Before we can design
a formal model of the planned state of the application area,
we must first understand and model its current state and
analyze it with regard to its accessibility to formalization.

Elicitation of the survey of the current state:
describe and model the existing socio-technical system:
– organization level (social IS, lock) and
– IT level (if there already is any technical IS)

Analysis of the current state:
– Is the lock pre-formalized (straight) or not (crooked)?
– How, to what extent can the lock be formalized (straightened)?

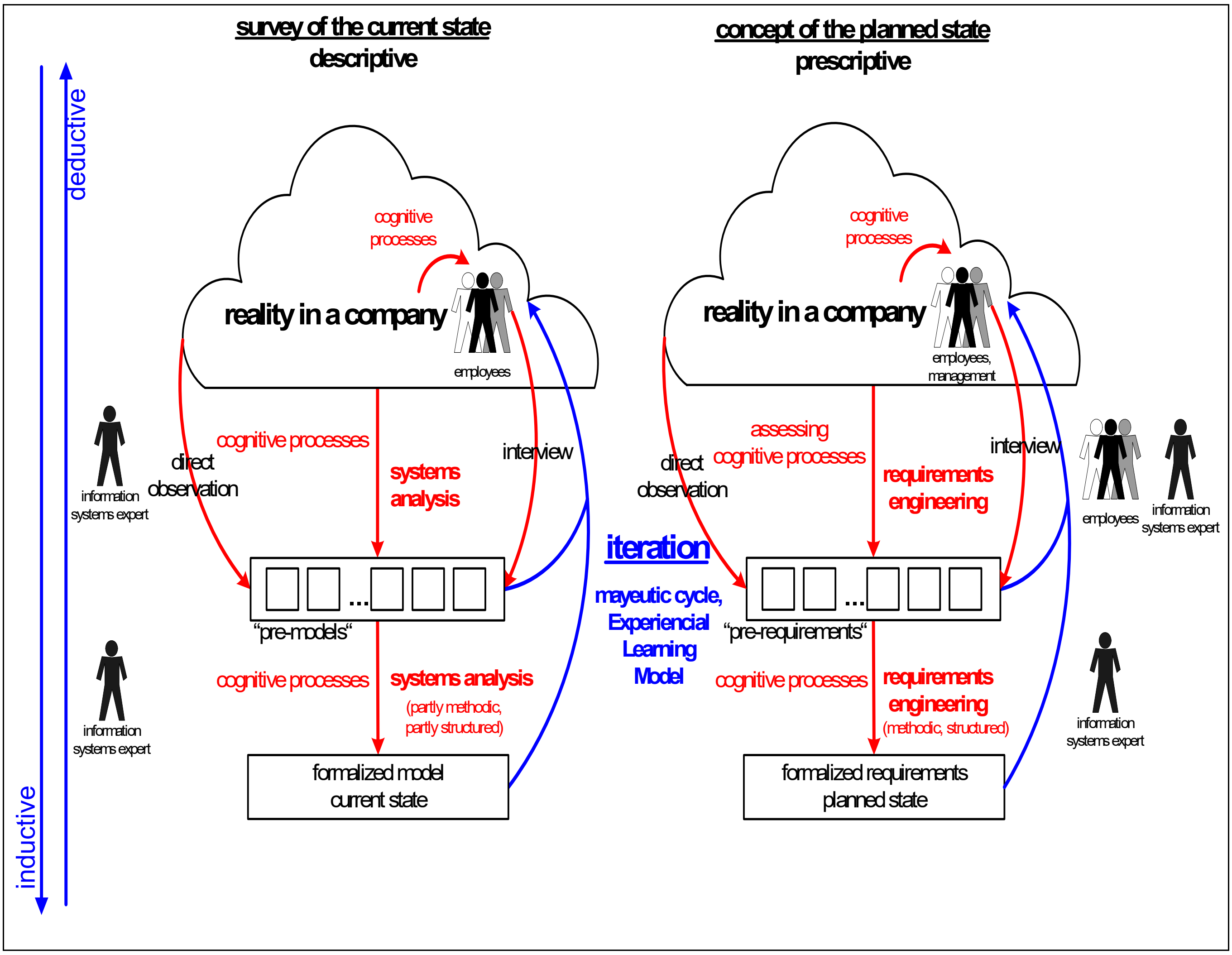
Regarding formalization, application areas of techn. IS differ in:
– pre-formalization
– potential for, accessibility to, suitability for formalization
(cf. deterministic vs. (non-) deterministic, chaotic domains)
– effort of formalization
→ not pre-formalized, scarcely formalizable object domains
→ partly pre-formalized object domains: implicit formal models
→ well pre-formalized object domains: explicit formal models

The design of descriptive models requires participative strategies.

Methods for the elicitation of the survey of the current state:
systems analysis on the basis of epistemology, systems theory

2 How to systematically design a technical IS 6

2.2-2.3 Details of modeling technical IS

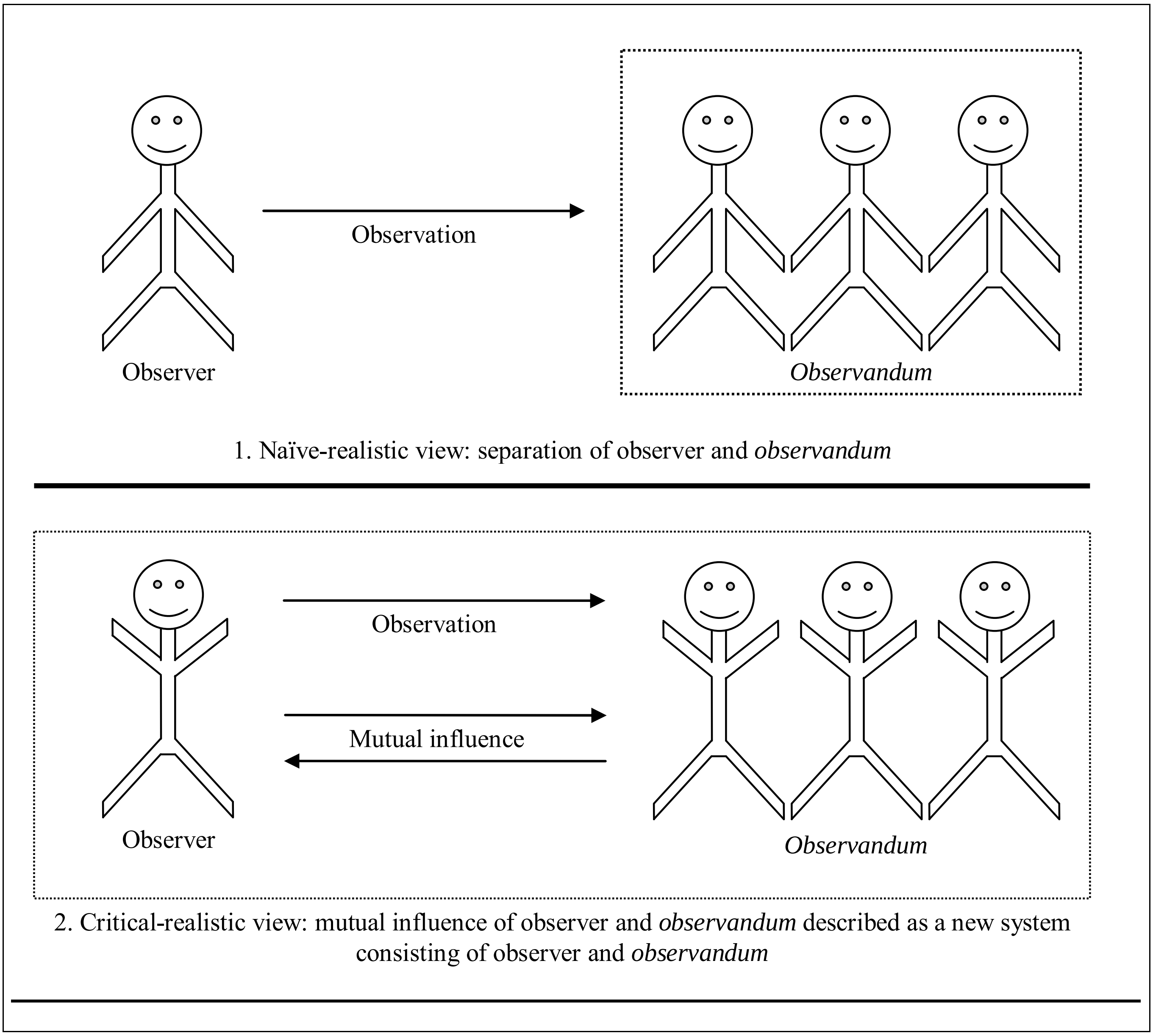


Cognitive processes in modeling technical IS
(Holl / Maydt, Epistemological foundations of RE, 2007, 53)

→ more in slides on empirism (mayeutic cycle)
The process of designing models of technical IS is a lot more complex than demonstrated up until now.
See advanced courses on IS.
E.g., there are different design techniques, such as top-down, inside out, view integration, umbrella models.
See course on software engineering.

2 How to systematically design a technical IS 7

2.2-2.3 Details of modeling technical IS Mutual influence observer – observandum



(Holl / Paetzold / Breun, Cooperative cyclic-iterative knowledge gain in IS anti-aging, 2011, fig. 3)

2 How to systematically design a technical IS 8

2.4 The two parts of the technical IS level

We now have the necessary prerequisites
to produce a technical IS.

The design of the technical IS level (key) consists of two levels:
– information-relevant level ~ business concept
– implementation-relevant level ~ IT concept
cf. phase concept / software process model

lock: organization level	
	key: information-relevant level
	key: implementation-relevant level

The business concept has to be modeled
in close cooperation between IS experts and business experts.

The IT concept has to be derived from the business concept
by IS experts only.

2 How to systematically design a technical IS 9

2.5 Phase concepts; software process models; summary of 2.2-2.4

main phase	subphase	model level	model purpose
analytical phase: problem analysis	elicitation of the current state of the soc-tech IS	information-relevant models	descriptive models (systems analysis, reverse engineering)
	analysis of the current state of the soc-tech IS		
	design of the planned state of the social IS (LOCK)		prescriptive models (require-ments engineering)
	design of the planned state (business concept) of the technical IS (KEY)		
synthetical phase: IT system development	design of the IT concept of the technical IS	implementa-tion-relevant models	
	programming		
	test		
	use	information-relevant models	
maintenance			

2.5 Unprofessional work without software process models

About German public projects going wrong:
Mertens, Peter: Schwierigkeiten mit IT-Projekten in der öffentlichen Verwaltung.
Informatik-Spektrum 35(2012) 433-446

Toll Collect (motorways)	2 years late
Health insurance card	permanent changes and delays
Administration of unemployed people data	hopeless
ELENA payroll data transfer	cancelled (data privacy)
eBalance: XBRL-based company profit data transfer	3 years late
Assignment of study places	at least 3 years late
Berlin airport	at least 3 years late

2 How to systematically design a technical IS

2.5 Software process models and information systems architecture concepts

Diaphasic multi-perspectivity

On its way through a systematic phase concept –
through a software (development) process model,
a model of a technical IS has to be transferred in several steps via different models,
each of which in turn is split vertically and horizontally,
from an organization / enterprise model on the information level to a technical model on the implementation level.

Every software process phase represents a certain perspective.

2 How to systematically design a technical IS

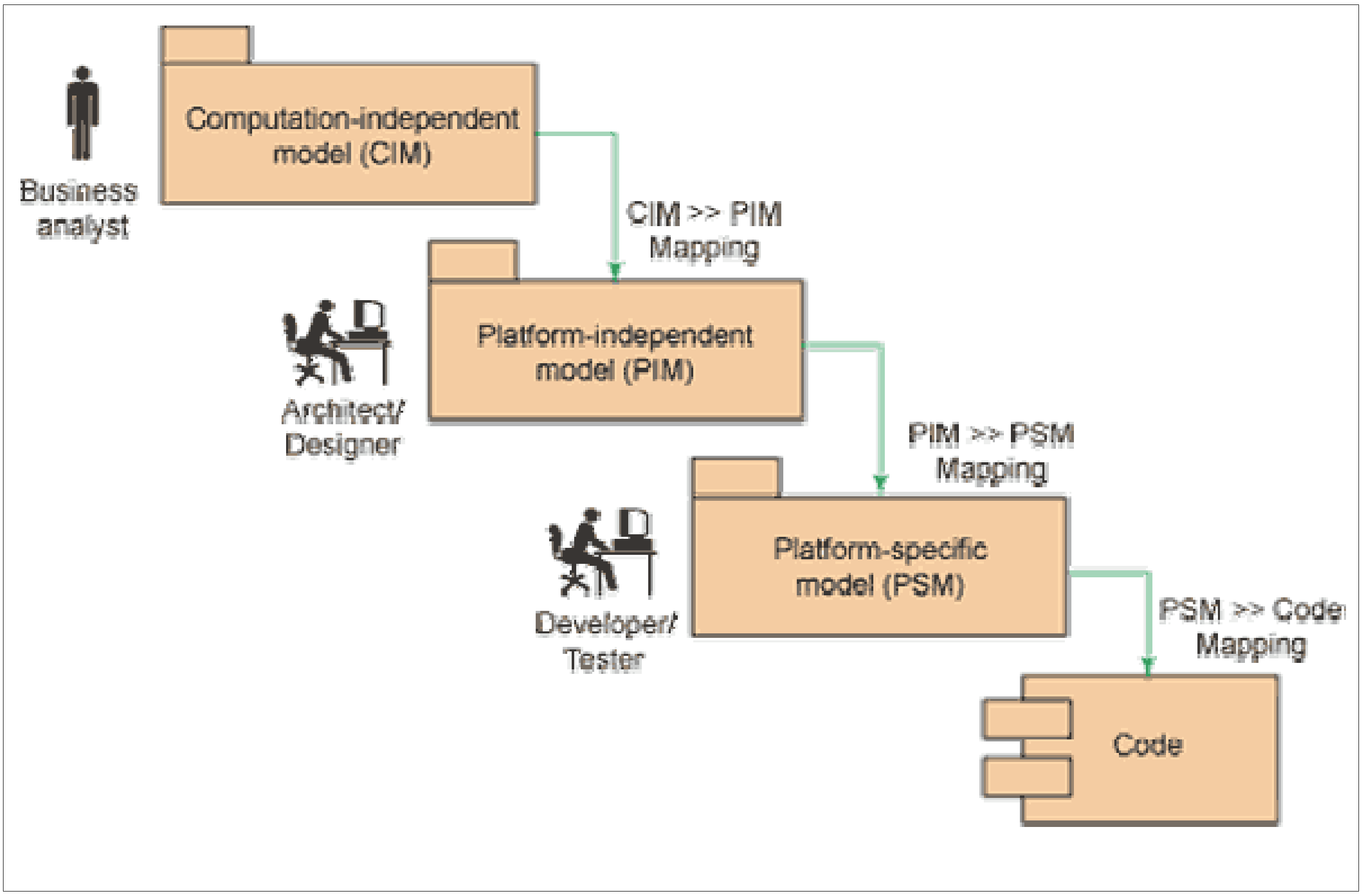
2.5 Software process models and information systems architecture concepts

Subphase	ARIS	MDA (Model Driven Architecture) by OMG (Object Modeling Group)
Design of the planned state / business concept	Business concept	Computation-independent model (CIM)
Design of the technical concept indep. of tools	IT concept	Platform-independent model (PIM)
Design of the technical concept dep. on tools	Implementation	Platform-specific model (PSM)
Examples	Siemens Systemhaus	Siemens Amberg (M^3 by MID Nürnberg) Application-specific model (ASM)
Tools	ARIS Toolset	Innovator (MID)

2 How to systematically design a technical IS

2.5 Software process models and information systems architecture concepts

Model-Driven Architecture (MDA)
by Object Management Group (OMG)



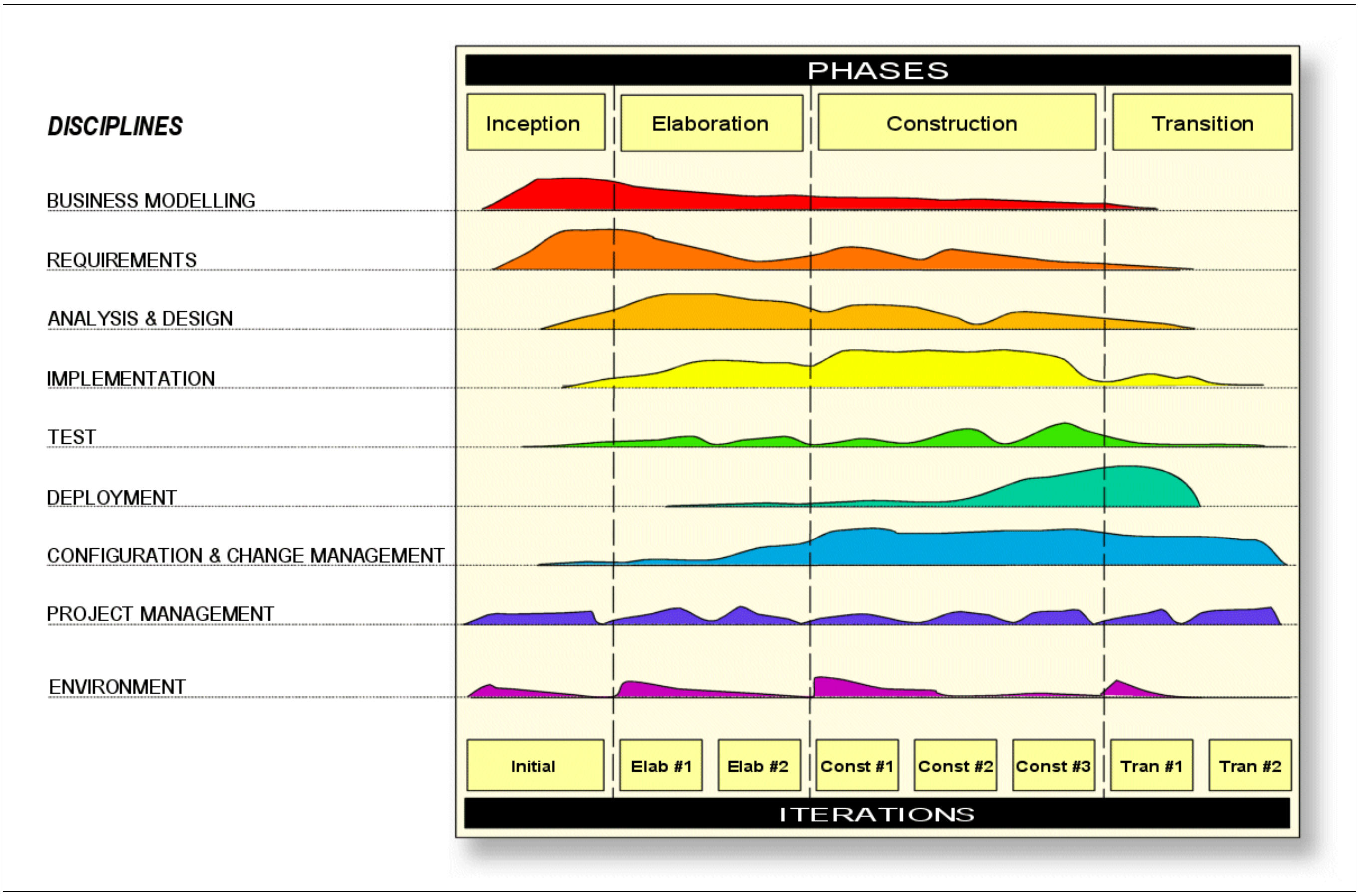
(Journal of Object Technology 2006, http://www.jot.fm/issues/issue_2006_03/column4/images/figure3.gif)

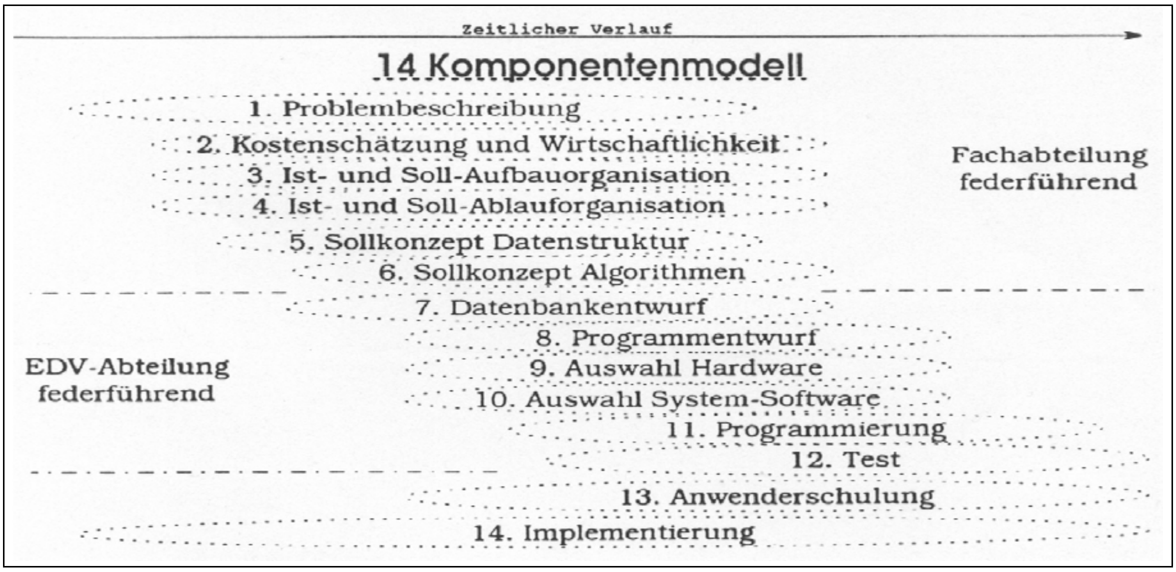
2 How to systematically design a technical IS

2.5 Software process models

Rational Unified Process (RUP)

by Grady Booch, Ivar Jacobsson, Rumbaugh

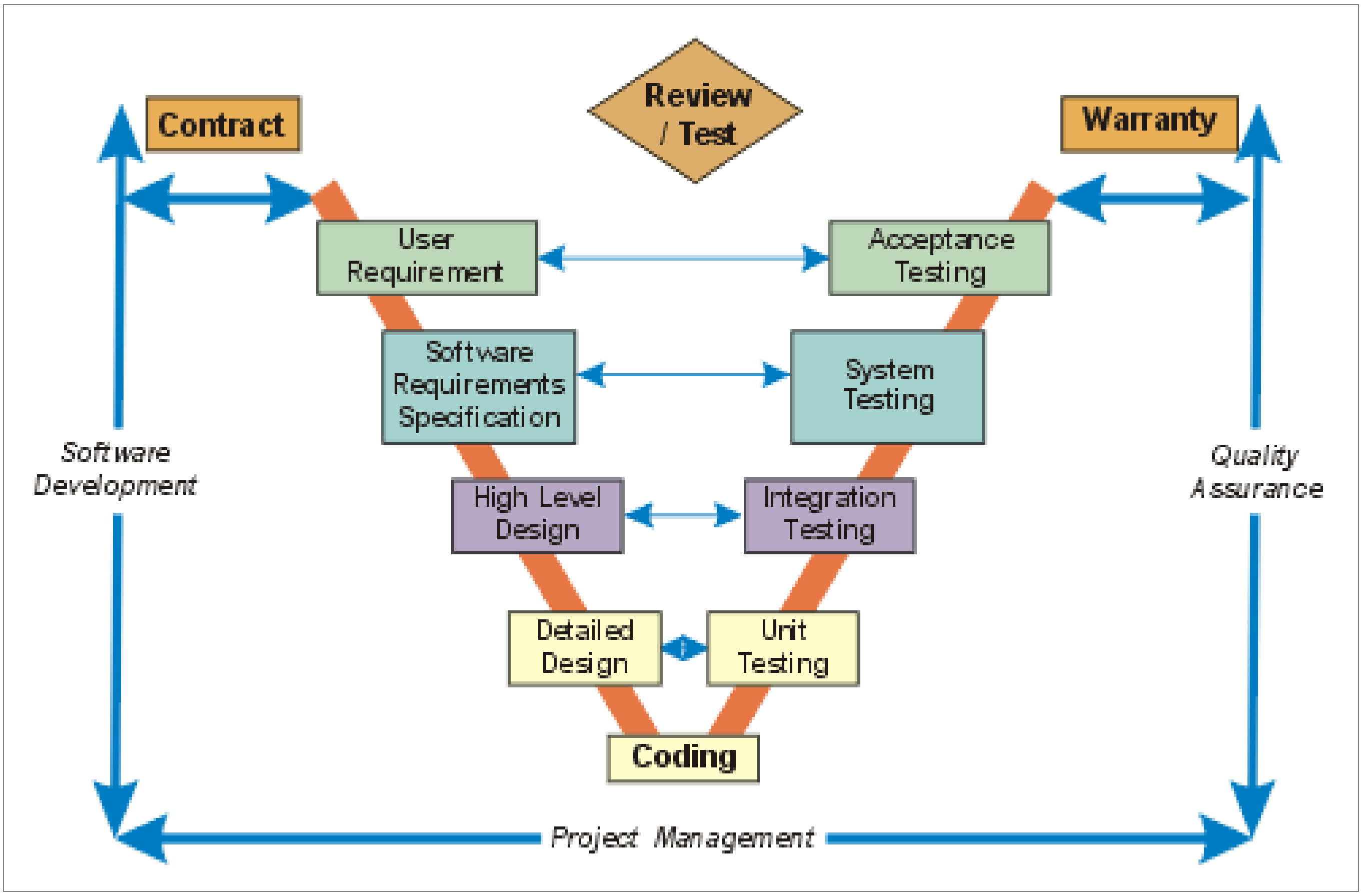




2 How to systematically design a technical IS

2.5 Software process models

V-Model

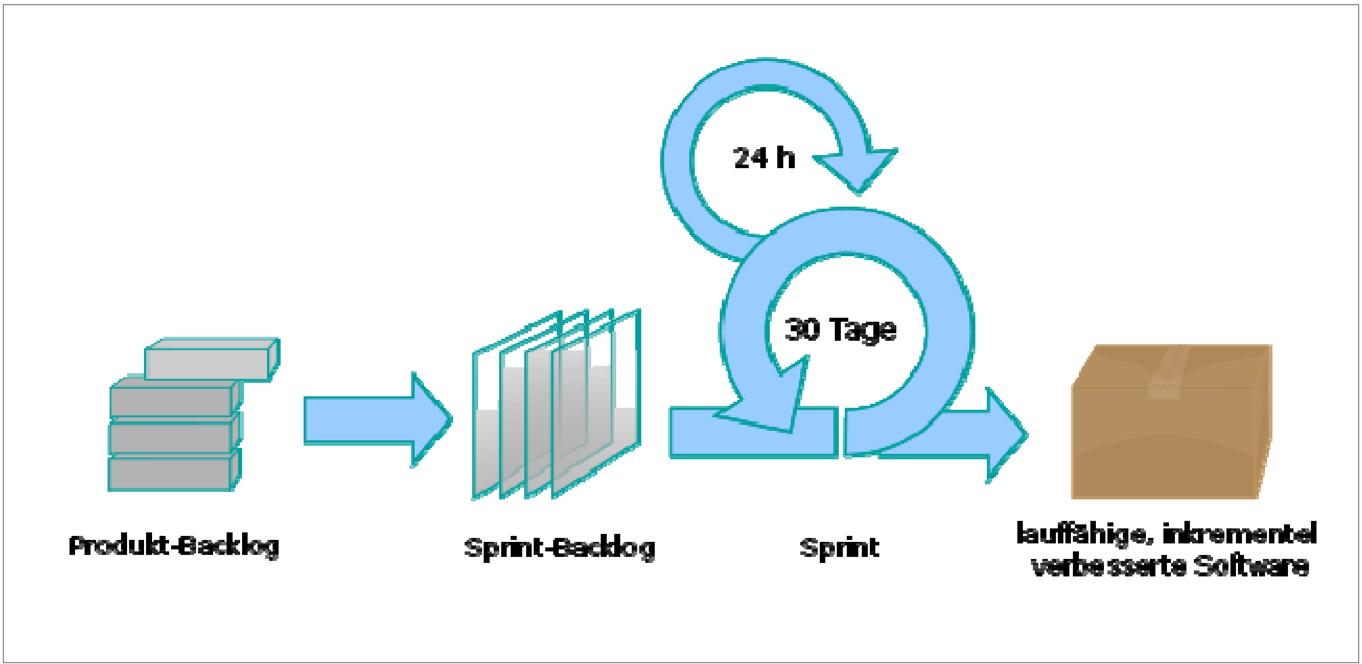


2 How to systematically design a technical IS

2.5 Software process models

Scrum process, agile software development

kleine Schritte:	**sprints**
Zwischenergebnisse:	**product increments**
Langfristplan:	**product backlog**
Kurzfristplan:	**sprint backlog**

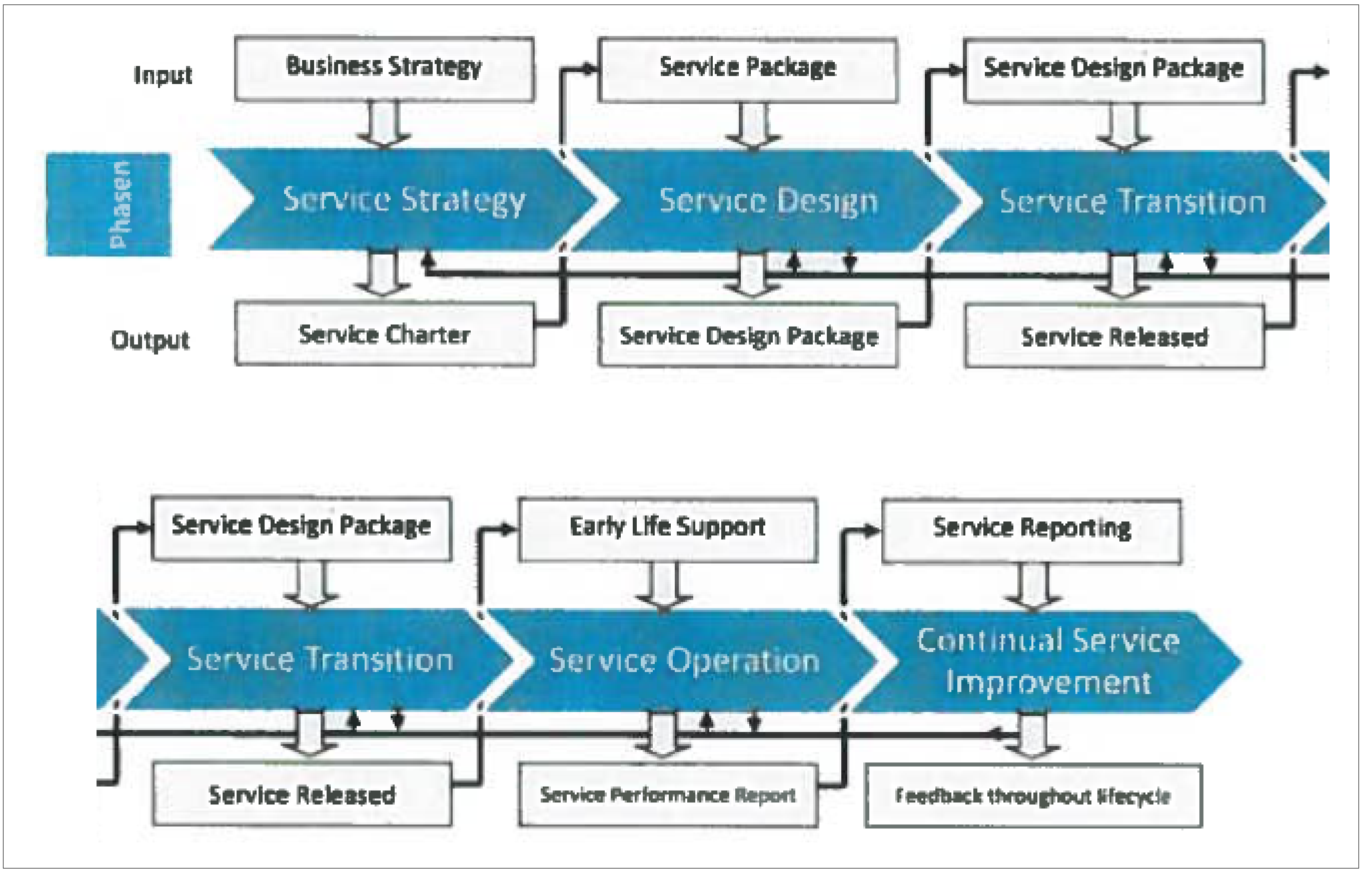


(wikimedia commons)

2 How to systematically design a technical IS

2.5 Software process models

ITIL (infrastructure library) Best Practices
für ITSM (service mgmt.)



(Günther 2012)

2 How to systematically design a technical IS

2.5 Phase concepts; model purposes and levels

Phase concepts or software life cycle models are process models for the professional, systematical development of software in well-defined steps (phases).

1 Model purposes:

– **descriptive**: **models of ...**
describe and analyze a current state: segment of reality
– **prescriptive**: **models for ...**
design a planned state:
application area in an organization and technical IS
present the desired information processing (test models)

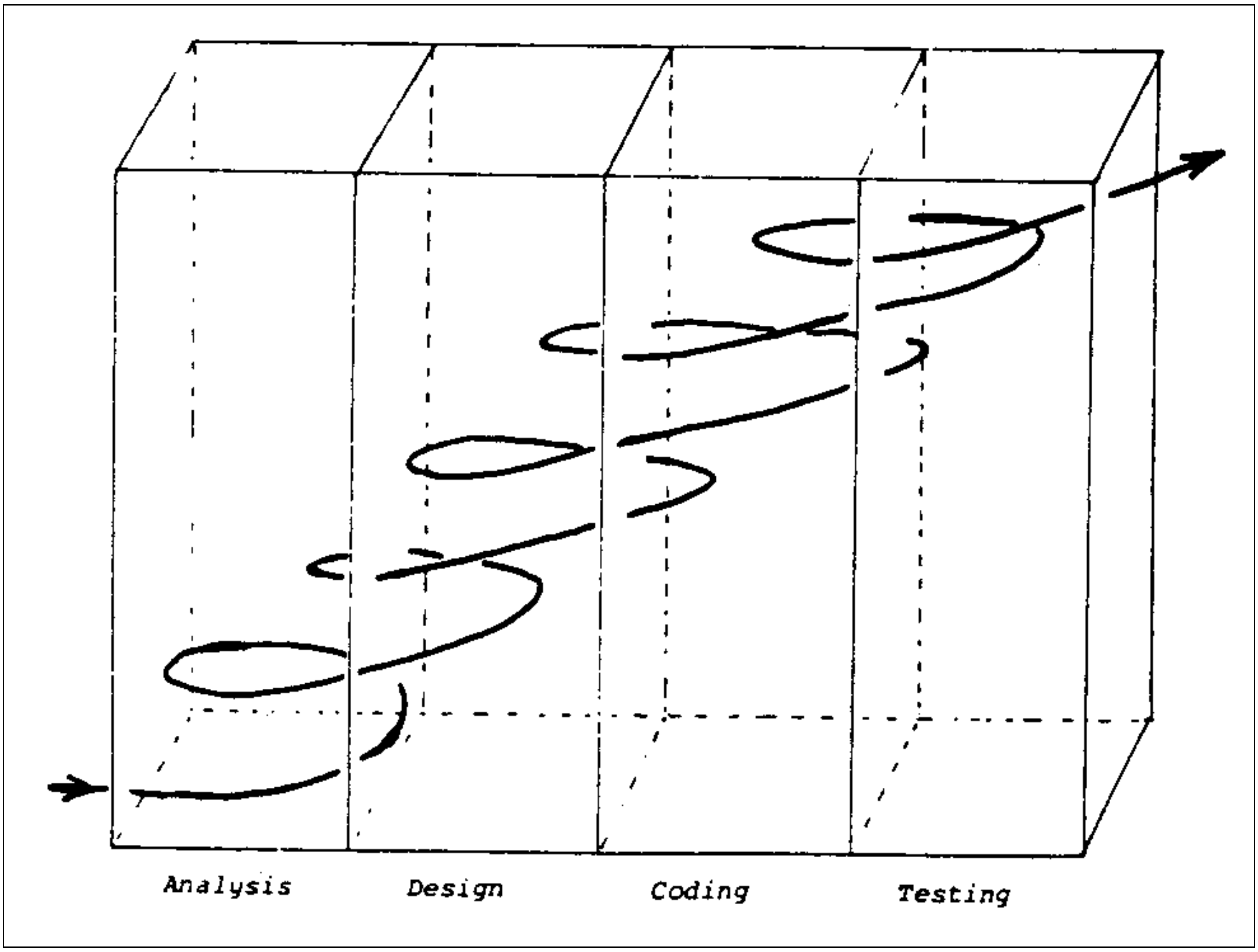
2 Model levels corresponding to main phases in a software life cycle:

– **models irrespective of IT details:**
analytical phase,
information-relevant level:
aims at the detailed understanding of a problem and the construction of a conceptual model

– **models with respect to IT details:**
synthetical phase,
implementation-relevant level:
aims at the construction of an IT system

2 How to systematically design a technical IS

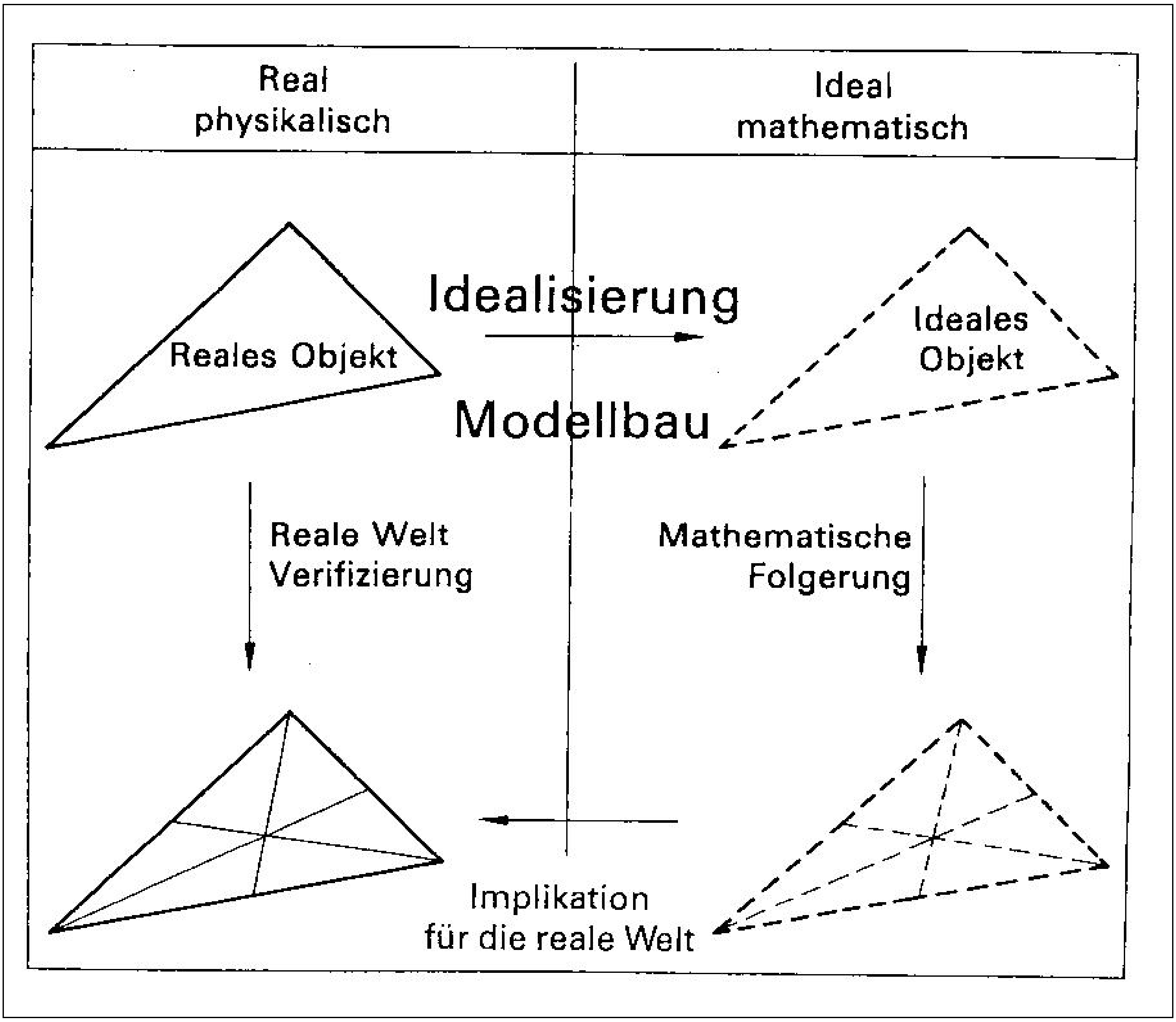
2.5 Phase concepts – simplification 1



Phase concept with loops, but without maintenance
(Hofstetter, SW-Entwicklung und human factor, *, 50)**

2 How to systematically design a technical IS

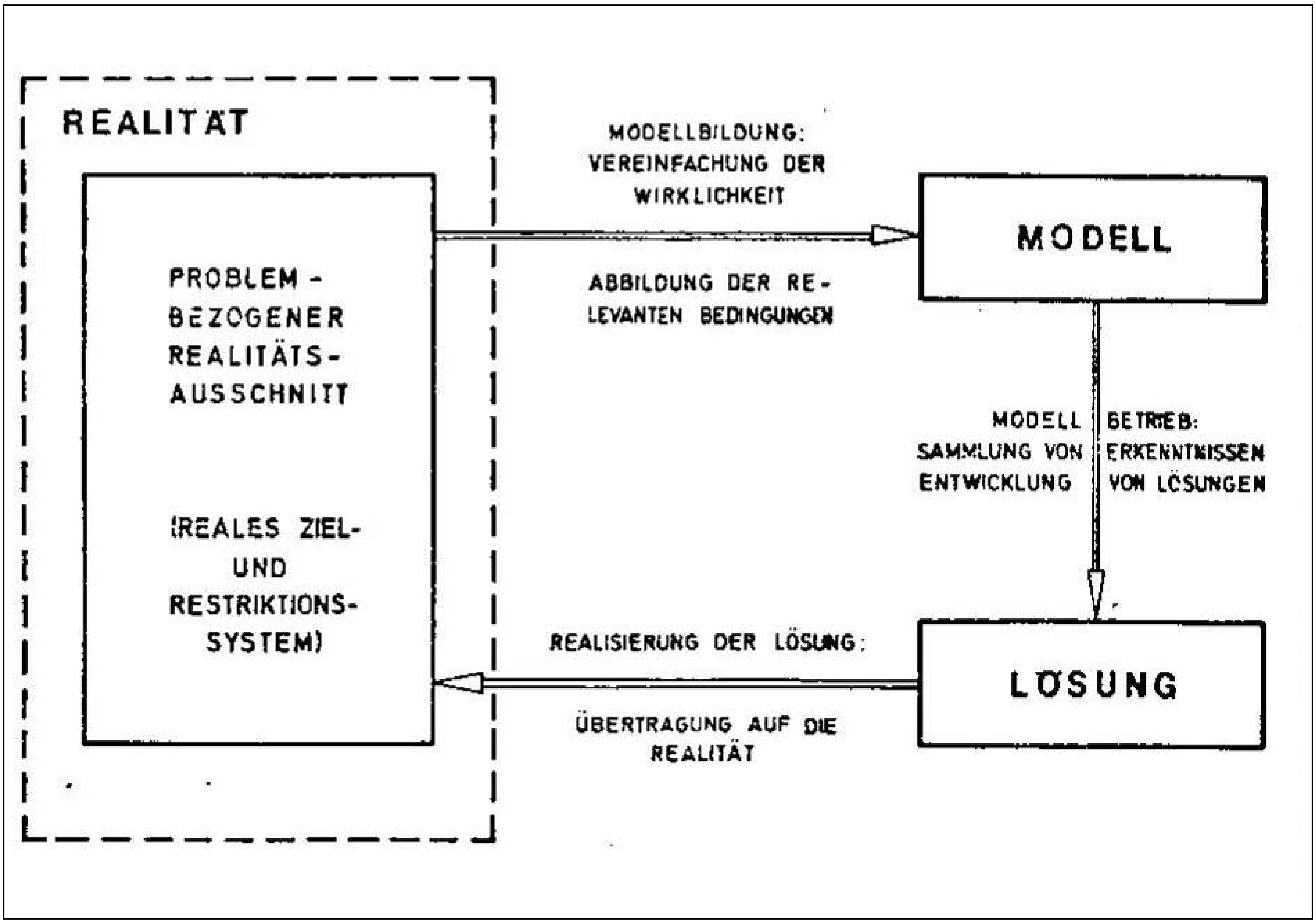
2.5 Phase concepts – simplification 2



Simple mayeutic cycle
(Davis / Hersh: Mathematical experience, 1994 [1981], 131)

2 How to systematically design a technical IS

2.5 Phase concepts – simplification 3



Simple mayeutic cycle
(reference ?)

2.6 How to change a technical IS – IS anti-aging, changed requirements management

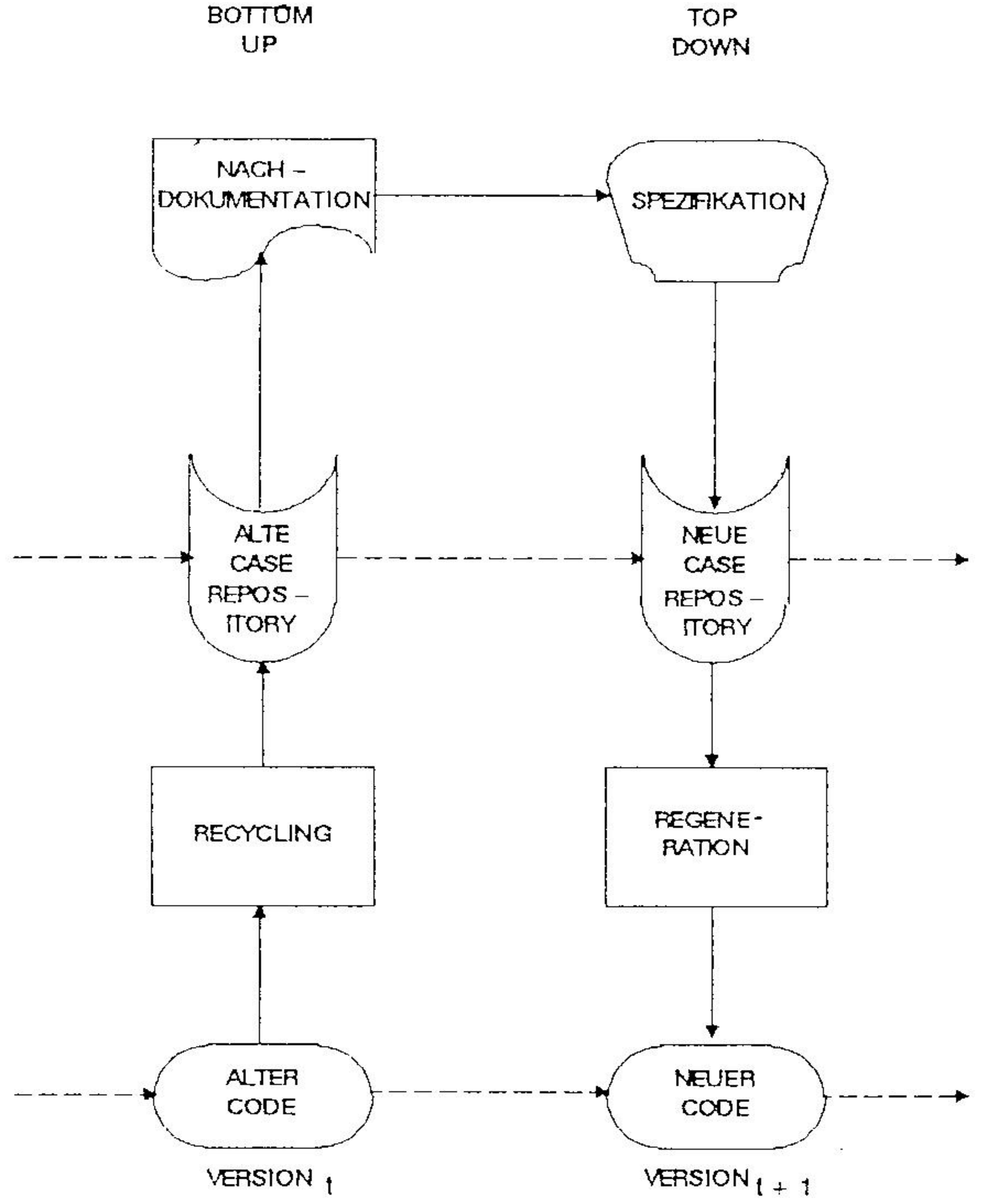


Abb. 2.10: Bottom-Up Software-Entwicklung

Bottom-up software development
(Sneed, Harry M.: Software maintenance, 1991, fig. 2.10)

3 Two sources for model construction 1

<table>
<tr><td>external
world
↓
World 1
objects of
cognition</td><td colspan="3">phenomenon,
individual experience
↓
World 2
knowledge of an individual
subject of cognition</td><td>model,
theory
↓
World 3
common
knowledge</td></tr>
<tr><td></td><td>perception,
cognitive
processes
(empiristic)
↓
reconstruct.
of World 1
→</td><td>memory</td><td>learning

rationalistic
↓
activations
of World 3
←</td><td></td></tr>
<tr><td>←</td><td colspan="3">↓
creation, induction
↓
design, influence ← new ideas, knowledge → publi-cation</td><td>→</td></tr>
</table>

Bi/trilateral linguistic sign

form, materialized signifiant	code of interpretation	meaning, signifié, W2|W3
object of cog.		

Model as complex bi/trilateral linguistic sign

materialized model repre-sentation	code of interpretation	model meaning W2|W3
object of cog.		

3 Two sources for model construction 2

Popper's World 1 (reality): empiristic method/approach
Organization, enterprise, department
observation and interviews (W3)
of domain experts by a model designer
(contrary to natural sciences: only observation)
preliminary description in pre-formal models: natural language
abstraction
check whether terminology is mathematically well-defined
final type construction
formalization (degree of pre-formalization is different)
reduction to axioms

often used for peripheral areas of models
often used for individual parts of an organization
(nominalist point of view: enumeration of individual objects)

Popper's World 3 (models, concepts, ideas): rationalistic method
reference models
activation in a model designer's brain
analogy-based transfer

often used for central areas of models
often used for standard parts of an organization, e.g. accounting
(universalist point of view: search for general principles)

Final step: integration of individual and reference models.

4 Empirism – Rationalism

Historically, there are
two different critieria for settling the truth of statements:
– naive empirism: experience and induction
– naive rationalism: reason and deduction

natural sciences	humanities
empirism Aristotle (384-322) John Locke (1632-1704) David Hume (1711-1776) John Stuart Mill (1806-1873)	rationalism Socrates (470-399) Platon (427-347) René Descartes (1596-1650 Sth) Baruch Spinoza (1632-1677) G. W. Leibniz (1646-1716)
Immanuel Kant (1724-1804): – synthesis of empirism and rationalism, – transcendental epistemology Konrad Lorenz (1903-1989): – evolutionary epistemology	

The mayeutic cycle contains empiristic and rationalistic parts,
that is, <u>observations and theories mutually influence each other</u>:
– Observations (experiences) change observation frameworks.
– Observation frameworks (intellect) exert an influence on
the selection of observation objects and
on observation interpretations.

5 References

pdf-files of my own publications: see my homepage.

Holl, Alfred:
Empirische Wirtschaftsinformatik und evolutionäre Erkenntnistheorie [Information systems as an empirical science and evolutionary epistemology].
In: Becker, Jörg et al. (ed.): ***Wirtschaftsinformatik und Wissenschaftstheorie. Bestandsaufnahme und Perspektiven*****.**
Wiesbaden: Gabler 1999, 163-207, ISBN 3-409-12002-5.
English translation on my homepage.

Holl, Alfred; Krach, Thomas:
Ubiquitäre IT – ubiquitärer naiver Realismus [Ubiquitous IT – ubiquitous naive realism].
In: Britzelmaier, Bernd et al. (ed.): ***Der Mensch im Netz. Ubiquitous Computing. - 4. Liechtensteinisches Wirtschaftsinformatik-Symposium an der FH Liechtenstein*****.**
Stuttgart: Teubner 2002, 53-69, ISBN 3-519-00375-9.

Holl, Alfred; Maydt, Dominique:
Epistemological foundations of requirements engineering.
In: Erkollar, Alptekin (ed.): ***Enterprise and business management. A handbook for educators, consulters and practitioners*****.**
Marburg: Tectum 2007, 31-58;
short version = contribution to:
Requirements Days 2006**, Nuremberg/Germany.**