Alfred Holl

Problem analysis: the analytical phase in a phase concept of IS development (SW process)

1 Motivation: phase concept (precise examination)

2 Preliminary examination of the application area (before phase concept / software process) (ITIL: strategic phase)
   2.1 Principles
   2.2 Problem of isolation / separability (system definition)
   2.3 Problem of pre-formalization

3 Precise examination of the application area (phase concept / software process)
   3.1. Principles
   3.2 Requirements engineering
   3.3 Requirements changes during problem analysis

4 Software evolution – IS anti-aging
## 1 Motivation: phase concept / software process (precise examination)

<table>
<thead>
<tr>
<th>main phase</th>
<th>subphase</th>
<th>model level</th>
<th>model purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>analytical phase: problem analysis</td>
<td>elicitation of the current state of the organization</td>
<td>information-relevant models</td>
<td>descriptive models (systems analysis)</td>
</tr>
<tr>
<td></td>
<td>analysis of the current state of the organization</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>design of the planned state of the organization (LOCK)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>design of the business concept of the technical IS (KEY)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>synthetical phase: IT system development</td>
<td>design of the technical concept of the technical IS</td>
<td>implementation-relevant models</td>
<td></td>
</tr>
<tr>
<td></td>
<td>programming</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>test</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>use</td>
<td>information-relevant models</td>
<td></td>
</tr>
<tr>
<td>maintenance</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
2 Preliminary examination (strategic phase)

2.1 Principles

Objectives:
- to figure out goal and purpose of the project
- to design a coarse business concept (Grobkonzept)
- to examine the feasibility (Durchführbarkeitsstudie)
- to finally decide about project initiation (Projektanstoß)

Coarse business concept: What is the problem / application area?

System definition: remember that organizations are open socio-technical information systems!

1. System delimitation, system boundaries, external coupling to the system surroundings (Umweltkopplung zur Systemumgebung):
- Remember: magnifying glass (geographic lens):
  model boundaries wider than direct IT application area
- Attention: cognitive processes during segmentation
- Attention: arbitrary object boundaries.
  Where is the rim of an organization?
  → 2.2 Problem of isolation / separability

2. Components, interactions, internal linkage (Binnenkopplung), partial systems on the next lower abstraction level

3. Goal and purpose of the system definition
Which purpose shall modeling during the precise examination meet? Which aim shall it reach?
Which question shall it answer?
E.g. preparation of the deployment of individual SW, selection of standard SW, planning of business process reengineering
2 Preliminary examination (before SW process)

2.1 Principles

4. Feasibility

4.1 mathematical-formal accessibility to modeling (important to estimate the duration of the project)
- pre-formalization (Vorformalisierung)
- accessibility to formalization (Formalisierbarkeit)
- effort of formalization (Formalisierungsaufwand)

→ 2.3 Problem of pre-formalization

4.2 technical (IT) feasibility
- coordination with existing IT systems
- individual SW, branch SW, standard SW

4.3 financial and temporal feasibility
- human and technical resources (Personal- und Sachmittel)
- possibility of financing, e.g. credits (Finanzierbarkeit)
- cost estimation (Aufwandsschätzung):
  - time, development costs, resulting costs (Folgekosten)

4.4 human resources psychology (Arbeitspsychologie)
- acceptance: hindered end-users (verhinderte Endbenutzer)
- resistance: forced end-users (gezwungene Endbenutzer)
- fear of rationalization (Rationalisierungsangst)

4.5 juridical and ethic feasibility
- technology assessment (Technikfolgenabschätzung)
- social acceptability/responsibility (Sozialverträglichkeit)
- data privacy (Datenschutz)
2.2 Problem of isolation / separability

**Observations**
trouble with not effective, isolated SW solutions
trouble with the limitation of systems

**Critical realism**
1. *complexity reduction* by creating segments and structures
2. there are natural *system-like structures*
   with strong internal and weak external connections
but: there aren’t any natural closed systems,
   only *open systems* [socio-technical information systems in IS]

=> systems: descriptive categories, not reality-immanent
   it is the observer who defines system boundaries

**There is no empirical knowledge without isolation of systems!**
(2nd epistemological dilemma)
Isolation is the pre-condition for
– the mere cognition of objects
– the transfer of feature sets, the perception of “gestalts”
– the comparison of objects
(Lorenz “The innate forms of possible experience” 1943: 319)

**Evolutionary epistemology**
1. cerebral cortex as carrier of cognitive processes
   has its origin in optical neural centers;
   consequence: *cognitive strategies are transferred* from
   primary objects of cognition
   visual-tangible (physical solids), simple,
   few interactions, ‘mesocosmic’
   to secondary objects of cognition
   socio-economical, sub-atomic particles, complex,
   numerous interactions, macro/microcosmical
2. *small segments* are better suitable as basis of analogy
2.2 Problem of isolation / separability

Consequences for IS
1. at least: SA context diagram or UML use case diagram with system surroundings and external connections
2. better: magnifying glass model: soft, blending system boundary with precision/magnification decreasing towards the rim
3. clear idea of the system’s purpose and objectives

Remark 1 (ambiguity of segmentation)
There are different ways of decomposing an object domain.

Example: magnet: optical field vs. magnetic field

Remark 2 (abstraction levels)
Humans cannot understand complex systems at first glance
===> complexity reduction by decomposition is necessary on different abstraction levels
===> problem of isolation occurs on every abstraction level

Remark 3 (process – system)
Processes can be interpreted as (linear) systems, therefore, there are equivalences
– open process ~ open system
– process boundary ~ system boundary (defined by observer)
– process decomposition ~ system decomposition (⇒ IV.3.1.2)
2.2 Problem of isolation / separability

Surrounding system and system surroundings
(from a diploma thesis)
2.2 Problem of isolation / separability

Accounts receivable system embedded in the enterprise
(Yourdon, Modern structured analysis, 1989, 336)
2.2 Problem of isolation / separability

System surroundings in a structured analysis context diagram
(from a diploma thesis)
2.2 Problem of isolation / separability

Constellations: form, where none exists, exemplified by a section of the northern celestial sky; to the right, the baroque representations by Cellarius 1708 (Riedl, Biology of knowledge, 1984, 157)
2.2 Problem of isolation / separability

Construction of structures by visual perception
(Eibl-Eibesfeldt, Biologie des menschlichen Verhaltens, 1995, 77)
2.3 Problem of pre-formalization, suitability for and effort of formalization

Observations
It is more difficult to model small organizations than large ones. SW development for accounting is easier than for production. Suitable descriptive categories are often unknown in organizations and model designers have to start from scratch to define them. It is difficult to estimate the time necessary for formal modeling.

Critical realism
With regard to formalization, IS object domains differ in:
- pre-formalization
- 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

Consequences for IS
Examine object domains with respect to 3 views of formalization. Respect the results in time and project management. Pre-formalized domains are starting points for IT (accounting). Don’t force formalization, allow chaotic oscillations (production). Check terminology used in organizations with regard to suitability for formalization.

Remark (structuring)
These considerations apply for pre-structures, suitability and effort for structuring as well.
3 Precise examination of the application area 1 (phase concept / software process)

3.1 Principles 1

1. **Elicitation of the current state (Ist-Aufnahme):**
   description of the lock

2. **Analysis of the current state (Ist-Analyse):**
   Is the lock straight (pre-formalized), crooked, can it be straightened?
   Analysis of defects (Schwachstellenanalyse):
   where are changes necessary?
   Requirements Engineering (continued in step 3)
   Proposals for solutions (Lösungsalternativen):
   how should changes be made?

3. **Concept of the planned state (Soll-Konzept):**
   formal model of the lock (organization)
   formal model of the key (technical information system)
   fine business concept (fachliches Feinkonzept, Pflichtenheft)

→ part 9: meta-models and notations of modeling approaches
3 Precise examination of the application area 2

3.1 Principles 2

Cognitive processes in IS modeling
(Holl / Maydt, Epistemological foundations of RE, 2007)
3 Precise examination of the application area 3

3.2 Requirements engineering: requirements in natural language

Rupp, Chris: Requirements Engineering. München 2001
based on R. Bandler/J. Grinder: Neuro-linguistic programming

This psychiatric approach 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?

**Generalization**
universal quantors:
  *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?
3.3 Requirements changes during problem analysis: temporal dynamics of the application area 1

Observations
SW does not meet requirements after long programming periods.

Critical realism
Every segment of the reality contains internal temporal dynamics, which can partly be deterministic and partly chaotic. Prognoses of its future behavior are only partly reliable, especially if a segment is disturbed.

Consequences for IS
in general:
1. keep your model of the planned state valid by quickly including changes in the application field in order to avoid using models of a past reality for programming
2. keep your models and your SW easily changeable in order to be able to easily change models and SW in the case of changes in the application field

Consequences for IS
in detail:
an aspect of changed / creeping requirements management overlapping phases and iterations in phase concepts dynamic design concepts permanent check of changes in the application field permanent contact to the future users participative strategies evolutionary SW development well-documented and easily adaptable SW some aspects of ‘extreme programming’ (user participation, quick development, small projects)
3.3 Requirements changes during problem analysis: temporal dynamics of the application area 2

Changed / creeping requirements gap
(Holl / Paetzold / Breun: IS anti-aging, 2011, fig. 17, p. 39)
4 Software evolution – IS anti-aging 1

Changes in reality require changes in software:

changed / creeping requirements management (cf. 3.3)

1st: introduction of changes in an IT independent (business) model

2nd: design of a new IT (dependent) model based on two sources:

1 the new IT independent model (information-relevant level)
2 the old IT (dependent) model (implementation-relevant level)

If the IT independent (business) model level is ignored and the new IT model is based only on the old IT model, SW complexity will soon increase a lot faster than the complexity of the underlying reality.

Reason: SW complexity has two sources:
– complexity of the underlying reality
– IT-immanent (IT dependent) complexity due to performance optimization, features of IT products etc.

Long-term SW evolution restricted to the IT level is impossible!

Include possible future changes in reality you can already foresee always in the IT independent (business) model level!

cf. Meir Lehman: E (embedded) type programs

requirements engineering on the basis of epistemology
4 Software evolution – IS anti-aging 2

Bottom-up software development
(Sneed, Harry M.: Software maintenance, 1991, fig. 2.10)
4 Software evolution – IS anti-aging 3

Increasing complexity of E-type systems
(Holl / Paetzold / Breun: IS anti-aging, 2011, fig. 20, p. 44)
Progressive and anti-regressive costs over time
5 References

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

Holl, Alfred:
Empirische Wirtschaftsinformatik und evolutionäre Erkenntnistheorie.
English translation on my homepage.

Holl, Alfred; Maydt, Dominique:
Epistemological foundations of requirements engineering.
short version = contribution to:
*Requirements Days 2006, Nuremberg/Germany.*

Holl, Alfred; Paetzold, Felix; Breun, Robert:
Cooperative cyclic-iterative knowledge gain in IS anti-aging.
Nuremberg: University of Applied Sciences 2011.