Alfred Holl

Meta-Models of IS Modeling Approaches

The primary focus are principles of modeling and basic elements of models independent of their graphical representation.

Only the secondary focus are individual notations of model representations (semantic networks with nodes and arcs / edges), such as Jackson, SA, UML etc.

0 Aspects of IS models and their notations (multi-perspectivity)

1 Rules for graphical model representations

2 Static function(-oriented) models: function structure models

3 Dynamic data(-oriented) models: information flow models

4 Static data(-oriented) models: data (structure) models

5 Dynamic function(-oriented) models: control flow models

6 Conclusion

0 Aspects of IS models and their notations

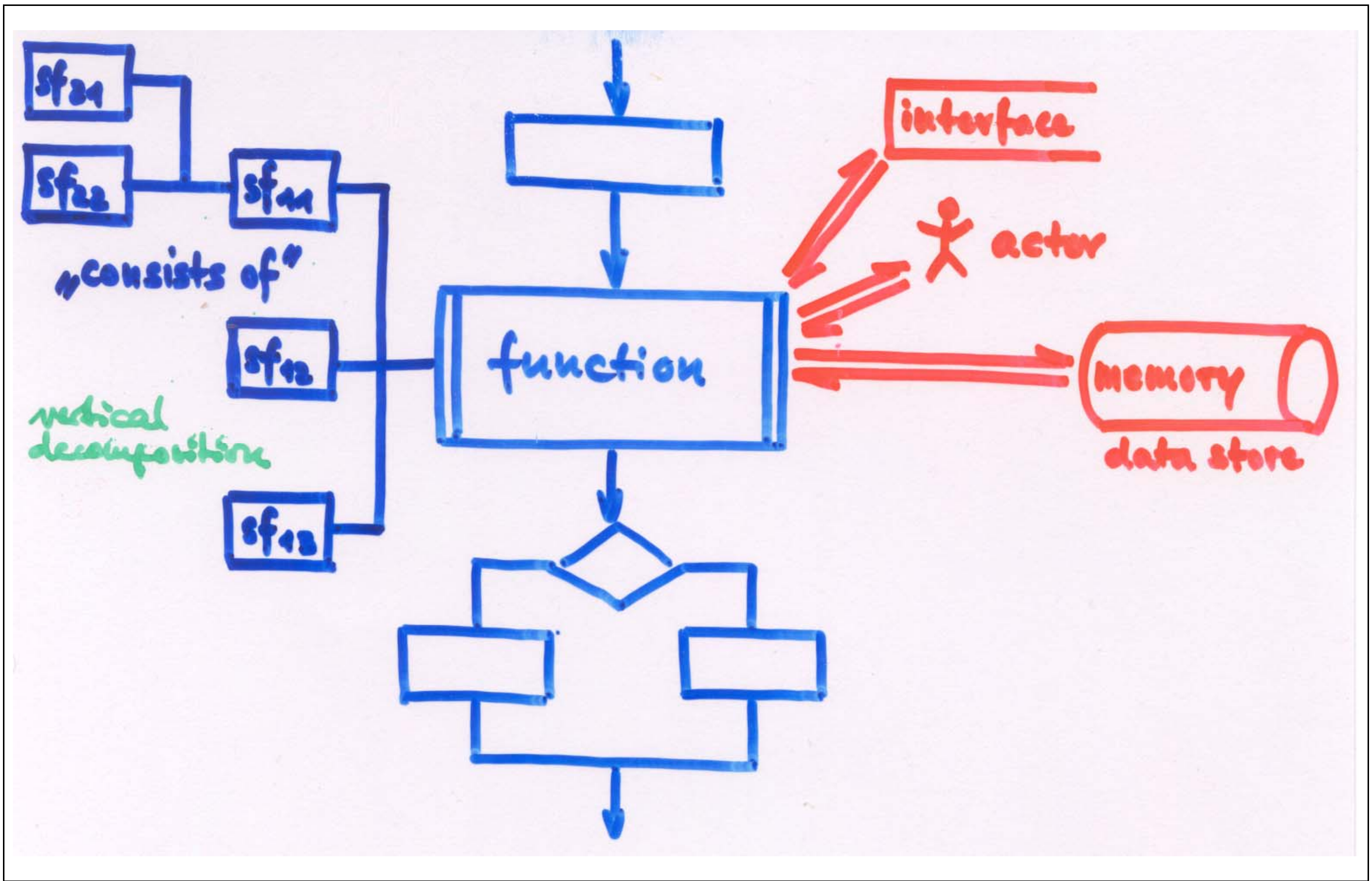
Horizontal multi-perspectivity / decomposition: static and dynamic data and function models 1

	static/structure models	dynamic/behavior models
data models	data (structure) models: data structure diagrams; entity-relationship models (ERM) UML class diagrams	information flow models: information / data flow charts / diagrams; Structured Analysis (SA); UML use case diagrams
function models	function structure models: compositional function trees; Jackson trees	control flow models: algorithms (functions); Nassi-Shneiderman diagrams, block diagrams (flow charts); business process models; UML activity diagrams; (UML sequence diagrams)

Each of the four aspects represents a certain perspective.

0 Aspects of IS models and their notations

Horizontal multi-perspectivity / decomposition: static and dynamic data and function models 2



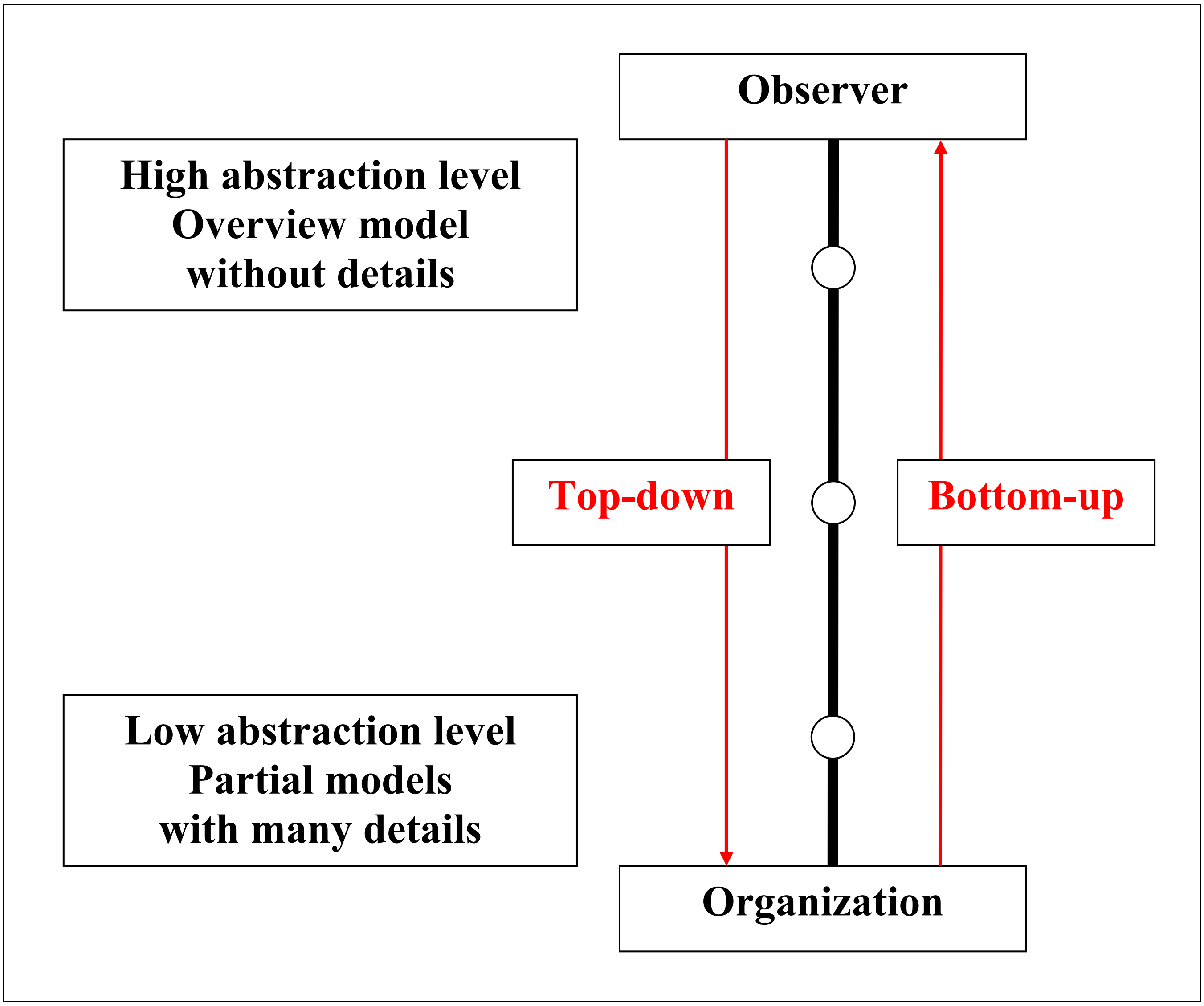
Static function model: function structure model irrespective of tests, iterations, sequences	Dynamic function model: control flow model	Dynamic data model: information flow model	Static data model: data structure model

0 Aspects of IS models and their notations

Vertical multi-perspectivity / decomposition: levels of abstraction

Using design methods (top-down, bottom-up, inside-out), models have to be decomposed into small and transparent partial models on different levels of abstraction (hierarchical levels with different degrees of abstraction).

Every level of abstraction represents a certain perspective.



1 Rules for graphical model representations

All of the nodes have unequivocal (distinct) names.

Vertical multi-perspectivity

Hierarchic decomposition (top-down refinement)
of all structure components on different abstraction levels
→ compatibility of adjacent abstraction levels

Horizontal multi-perspectivity

→ compatibility of adjacent model aspects

---

7 to max. 9 nodes per diagram ('chunk', 'Superzeichen')
cf. Miller, George A.: *The magical number 7, plus or minus 2*, The Psychological Review 63(1956), 81-97

→ transparent arrangement of the symbols
→ sequence of comments according to this arrangement

Diagrams and comments mutually complete each other.

2 Function structure models

function trees, ‘consists of’-hierarchies

irrespective of control flow (conditions, iterations)

3.1 Information flow models: Elements 1

Nodes:

1. functions

– template for function names: verb + noun (object)
→ quantifier if singular and plural cannot be distinguished

2. stores, memories

Types of information media (‘Datenträger’; to be marked in diagrams):
– digital stores / media
– paper (printout, document (‘Beleg’, ‘Schriftstück’), card index (‘Kartei’), binder (‘Ordner’))
– mental memories, skills, knowledge (head)
→ type purity (only one information type in every store)
→ type compatibility of the attached information flows
→ type conversion only with functions

3. actors (intra-system and extra-system (SA -interfaces); extra-system ones can be intra- or extra-organizational):
function owners, responsible persons (‘Aufgabenträger’, ‘Funktionsträger’), persons involved (‘Beteiligte’); institutions; IT systems

3.1 Information flow models: Elements 2

Arcs:

data / information flows (type purity)
– between functions and stores / actors
→ not between different functions
– distinction analogous to store types
(digital, paper, oral communication)
– flow direction (cf. store access modes read / write)
if necessary material and money flows

Not: processes, procedural structures
Auxiliary approaches:
– graphic arrangement of the function symbols
– [event flows (a second arrow type; cf. CASE4.0)]

3.2 Information flow models: Notations

Traditional data flow chart

Process matrix (data flow chart in the form of a table):
– functions, stores, access mode

Structured Analysis (SA) according to Tom de Marco 1978:
– level 0: context diagram (system delimitation)
– only extra-system (external) interfaces / actors

UML use case diagram (‘Anwendungsfalldiagramm’)
– intra-system and extra-system actors

4.1 Data structure models: Elements
1 Entity / object types 1

Nodes:

Entity types, object types, (object) classes

1. attribute structure (attribute ≈ property dimension)
– attributes of the objects (instances) of a class (Instanzattribute)
– attributes of the corresponding class (Klassenattribute)

2. attached procedures, elementary functions, services, ‘methods’
(called by messages): data encapsulation
– services of the objects (instances) of a class (Instanzmethoden)
– services of the corresponding class (Klassenmethoden)

3. object types
– transient: exist during run-time only
– persistent: stored in permanent memories

→ individual elements: entities, objects, object instances

Relation between SA and OO
– refine SA data stores to object types / classes
– refine SA functions to OO services

4.1 Data structure models: Elements
1 Entity types 2

Entity, object
not: a real object, but: a model, a tuple of attribute values

Entity structure: attribute structure
2 entities have the same structure iff they have the same attributes

Set of entities
Set of arbitrary entities with the same structure

The following problem leads to a definition of an entity type:
A customer table shall only contain one address for each customer, the current address.

Entity type: 2 properties
- **structure property: invariable attribute structure**
- **compatibility property: definition of a primary key**
  - **2 entities with the same structure are compatible iff they can be elements of the same entity set iff their primary key values are different (entity integrity).**

Entity set
Set of compatible entities with the same structure (table).

There are always several entity sets of an entity type.
Not every set of entities with the same structure is an entity set.

	World 1	World 3
single object	one real object	entity
set - type	set of homogeneous real objects	entity type

Data store (digital, paper, head): can contain several entity types

4.1 Data structure models: Elements
2 Relationships, associations 1

Arcs:

Relationships, associations (OO) regard
the interrelations between entity types from 3 points of view:

1. numeric classification: cardinality, multiplicity (OO)
– n:m many-to-many relationship
– 1:n one-to-many relationship
– 1:1 one-to-one relationship
– 1:c conditional relationship (c = 0, 1)
– special two-dimensional relationships such as c:c, c:n
– multidimensional relationships: n:m:p, n:m:p:q (e.g. time-table)

CAUTION: The cardinality is also a type
that is, it can have different values for individual entities.

Example:
A one-to-many relationship "customers → orders" can contain
– individual customers with many (two or more) orders
– individual customers with one order
– individual customers with no orders

2. syntactic description: primary key → reference key
– on entity type level: key attributes
– on entity level: key attribute values

→ Referential integrity required!

4.1 Data structure models: Elements 2 Relationships, associations 2

3. semantic interpretation

3.1 simple association without any semantic interpretation

Examples:
1:n customers – orders
1:c employees – (head of) – department

3.2 compositional relationship:
'consists of', 'whole-part', aggregation

Composition (special case of aggregation): existential dependence

Examples:
1:n orders – order lines
1:c car – air condition

4.1 Data structure models: Elements 2 Relationships, associations 3

3. semantic interpretation

3.3 taxonomic relationship:
‘is a’, generalization (umbrella term) / specialization
inheritance of attributes and services
polymorphism of services

Example:
1:c business partners – customers in combination with
1:c business partners – suppliers

Interpretation different in OO class diagrams:
no instances of abstract classes

general concept	**basic class abstract class**	**↑ generalization**
special concept	**derived class**	**↓ specialization (with inheritance)**

4.2 Data (structure) models: Special models and their notations

1. Pure data models in general

all features mentioned without services (attached procedures)

ERM: entity-relationship model (Chen) and its extensions

DSD: data structure diagram (Bachman) and
Oracle diagram:
no semantic interpretation of relationships

2. 3NF models: special pure data models reduced to axioms (controlled redundancy)

- **no services (attached procedures)**
- **no semantic interpretation of relationships**
- **one-to-many relationships only**

favorite diagram: DSD

3. Static object models

Thesis: should be based on 3NF data models in order to have a stable basis quite independent of subjective influence

all features mentioned
logical primary and reference keys are not always used

UML class diagram

5.1 Control flow models: Elements 1

In the following, behavior meta-models will be examined from the point of view of information systems.

That is, there will be a focus on the activity-on-node variant.

The activity-on-arc variant (state transition networks, Petri nets), which is important for theoretical computer science approaches, will be excluded.

5.1 Control flow models: Elements 2

Nodes:

1. function, action (computer-aided or not)
   function unit, function module

– name from the view of the organization
– decomposition-marker: reference to sub-processes
– algorithm, internal logic in a note
– duration, start time, end time
– features, feature values (→ theory of gestalt)
– IT support: computer-aided or manual

2. initiating and resulting events

3. actor: person/role/department responsible for the action
   partly connected with data flow

4. external (business/communication) partners
   connected with data flow

5. data stores accessed: input data and output data
   connected with data flow

6. resources used (machines etc.)

	World 1 (reality)	World 3 (model)
single object, "instance"	one individual course of events in an organization	business process instance
set - type of similar objects	set of homogeneous courses of events	business process type

5.1 Control flow models: Elements 3

Arcs:

1. control flow: temporal interrelation of functions
   (cf. structured programming)

– temporal succession: sequence (predecessors and successors)
  mandatory or arbitrary (pseudo-parallelism)

– condition: alternative, selection (IF, XOR)
  case discrimination (CASE)
  or complex rule (decision table)
  disjoint and complete (if incomplete a standard path)

– repetition: iteration, loop (WHILE or REPEAT)
  test-first loop and test-last loop

– recursion

– simultaneousness: parallel processing, parallelism (AND)
  mandatory or arbitrary (pseudo-parallelism)

CAUTION:
all control flow elements without the mere sequence must have
a divergent delimitor (begin) and
a convergent delimitor (end, synchronization);
the delimitors have to be arranged symmetrically in a diagram:
IF – ENDIF, CASE – ENDCASE, LOOP – ENDLOOP etc.

2. data flow (only partly)

3. mere connectors to actors and resources used

5.2 Control flow models: Special models and their notations 1

1. Classical notations

1.1 Traditional notations for structured programming

flow chart, block diagram ('Programm-Ablauf-Plan')
structure diagram, structogram (Nassi-Shneiderman diagram)
Jackson tree
- Jackson structured design (JSD)
- Jackson structured programming (JSP)

functions and control flow

1.2 Decision table

complex conditions and functions: rules

1.3 Network model(ing technique)

functions, sequence, parallel processing,
duration, start time, end time
→ critical path

1.4 Control flow plus data flow

HIPO: hierarchy plus input-process-output (Mills 1972, IBM)
functions, control flow, data stores, data flow

5.2 Control flow models: Special models and their notations 2

1.5 Swim lane diagrams

functions, control flow, responsible departments
predecessor of UML activity diagram
Arbeitsablaufdiagramm: Arbeitsschritte – Abteilungen
Organisationsprozessdarstellung (H. F. Binner)

2. Business process models

Event-driven process chain (EPC)
OMG Standard: Business Process Diagrams (BPD) using
Business Process Model and Notation (BPMN)

functions, control flow
events
actors, partners, data stores, resources, data flow
swim lanes (responsible departments)

3. Dynamic object models

UML activity diagram

functions, control flow
events (non standard)
actors, partners, data stores, resources, data flow
swim lanes (responsible departments)

UML sequence diagram

classes, elementary functions called by messages, control flow

6 Conclusion: Model notations as semantic networks

All the models mentioned can be represented by semantic networks with nodes and arcs.

Analogies:
data model: entity types and relationships
process model: functions and control flow
different types of control flow correspond to different types of relationships between entity types
in addition:
data flow models contain different types of nodes: functions, actors and data stores (entity types)

---

7 References

Böhm, Corrado; Jacopini, Giuseppe:
Flow diagrams, Turing machines and languages with only two formation rules.
***Communications of the ACM* 9(1966) 5, 366-371.**

Dijkstra, Edsger:
GOTO statement considered harmful.
***Communications of the ACM* 11(1968) 3, 147-148.**