

Business-ICT Alignment

Roel Wieringa

Center for Telematics and Information Technology
(CTIT),
University of Twente, the Netherlands
www.cs.utwente.nl/~roelw

Joint work with Henk Blanken, Pascal van Eck, Maarten Fokkinga,
Paul Grefen (now at TUE), Ander de Keijzer, all of CTIT.

Research partially sponsored by the Telematics Institute
<http://www.telin.nl/NetworkedBusiness/Graal/ENindex.htm>

1

Contents

1 Alignment	4
1.1 The Alignment Problem	4
1.2 State of Research	5
1.3 Our Research Goal	6
2 Research Method	7
3 Conceptual Framework	8
4 Findings	20
4.1 Applications and infrastructure	20
4.2 Conway's law	21
4.3 Noncommutative alignment	23

2

4.4 Application alignment	24
4.5 Infrastructure alignment	27
4.6 Coordination patterns	31

5 Further work	32
-----------------------	-----------

3

1 Alignment

1.1 The Alignment Problem

Alignment is the mutual adjustment of IT and business

- Practical problem (*action problem*)
 - IT does not satisfy business goals and vice versa
 - Mergers and reorganizations: Integration and reintegration
 - Legacy: More integration. Old technology never dies.
- Research problem (*knowledge problem*)
 - Lack of knowledge about relationship between business design and IT design

4

1.2 State of Research

- Studies of software architecture are too detailed to be of use to the practicing information systems architect
 - Patterns like MVC
 - ADLs
 - Architectural styles and software attributes (<http://www.sei.cmu.edu/>)
- Studies of business-IT alignment are too strategic to be of use for the practicing IS architect.
 - Strategic business objectives and IT objectives
 - Management support
 - Personnel policies

5

1.3 Our Research Goal

- Develop operational concepts and guidelines for business-IT alignment.
- Guidelines Regarding Architecture ALignment (GRAAL)

<http://is.cs.utwente.nl/GRAAL/>

6

2 Research Method

Case study research

- A *case* is an instance of a situation of interest.
- We study cases in which alignment decisions are made.
- In our case studies, we analyze architecture documents of an organization and obtain feedback from architects.
- Need a *conceptual framework* in terms of which to analyze different cases
- After each case study, generate and refine hypotheses.

7

3 Conceptual Framework

- Comprehensive analysis of existing frameworks in IS, Software Engineering, Systems Engineering, industrial product design [2, 3, 4].
- Validation on well-known examples.
- Application in M.Sc projects.
- Validation on real-life cases in large service organizations.

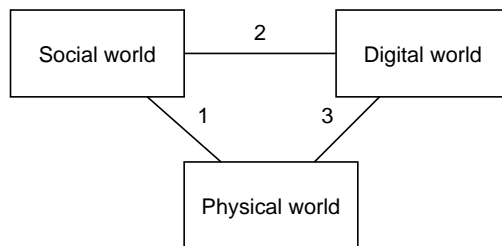
8

Three worlds

- *Social world*: The world of institutions.
 - E.g. employees, roles, business goals, business processes, money, income, expenses, cash flow, ...
- *Physical world*: The world measurable by the MKSA system: Meters, kilograms, seconds, amperes.
 - E.g. buildings, hardware, distances, locations, wires, electromagnetic waves, light waves
- *Digital world*: The world of physical symbol manipulation.
 - E.g. Documents (paper or electronic), dictionaries, software

9

Three alignment problems



1. Social function of buildings, rooms, locations; physical security; etc.
2. Meaning of data; function of software systems; digital security; etc.
3. Allocation of software to hardware, physical location of documents, etc.

10

Systems

- A system is any coherent collection of parts.
 - Social systems, digital systems, physical systems
- *Subsystem* is part of a system, considering all its properties
- *Aspect system* is the system, considering only some of its properties.
- Considering subsystems and considering aspect systems are two ways of reducing complexity.

11

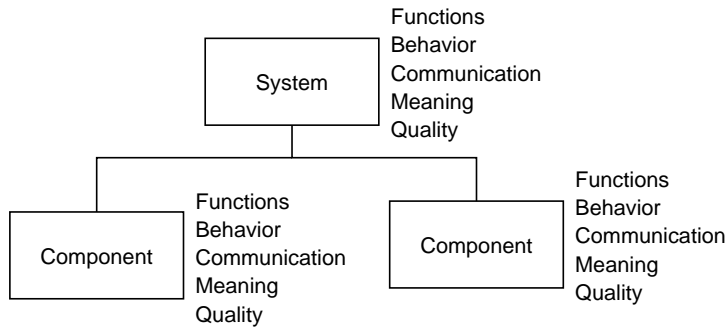
Aspects

Each system has external interactions with its environment.

- *Functions* are services provided by the interactions
- *Behavior* is ordering of interactions over time
- *Communication* is ordering of interactions in “space” (who talks with whom)
- *Semantics* is the meaning of the interactions
- *Quality* of the interactions is their utility
 - Usability
 - Performance
 - Reliability
 - Maintainability
 - Interoperability
 - ...

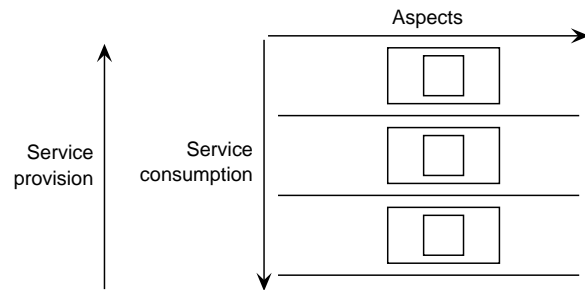
12

Aspects appear at every level of any aggregation hierarchy



13

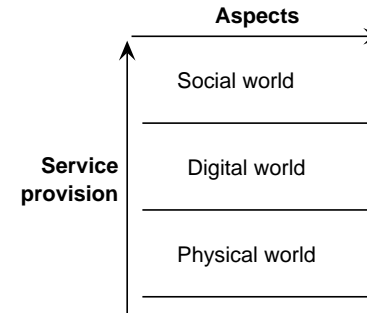
Layering



- Entities at lower layers provide services to entities at higher layers.
- At each layer, entities have internal components.
- At each layer, entities have functions, behavior, communication, meaning, quality aspects

14

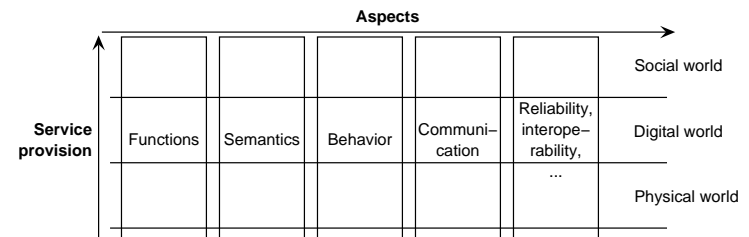
Putting it all together: Adding the three worlds



- This is a simplification: There are also service provisions from top to bottom, that we ignore now.

15

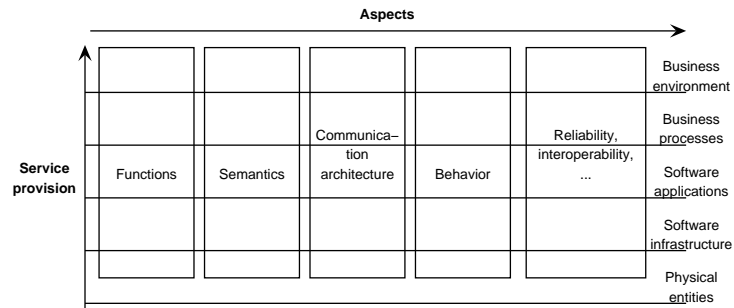
Putting it together: Adding aspects



- Semantics: This is the information aspect at every layer (even the physical).

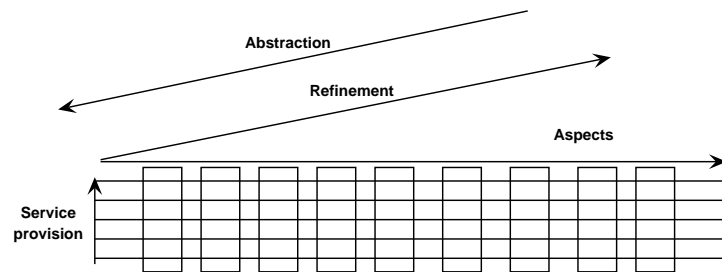
16

Refining the layering



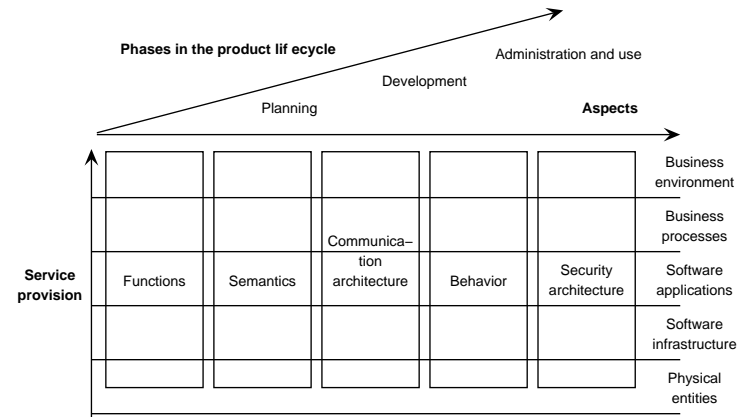
- Infrastructure: operating systems, middleware, DBMS, WFMS, ERP and other systems for general use.

The refinement dimension



- Refinement (adding details) is the inverse of abstraction (removing details).
- A property of descriptions, not of entities.

The temporal dimension



- Each entity may have several versions concurrently.

4 Findings

4.1 Applications and infrastructure

Application:

- Supports particular business process
- Particular user group
- Development is event-driven
- Evaluation is local

Infrastructure:

- Generally available
- No particular user group
- Development is time-driven
- Evaluation is global

4.2 Conway's law

Conway:

- A system whose architecture is too big to fit one head, is designed by a group.
- The architecture of such a system is isomorphic to the architecture (work breakdown structure, communication pattern) of the design group.
- The architecture will change.

We observed this for architecture groups:

- Tacit domain knowledge of architects assigned to user group
- Early warning system for changes
- Effective requirements engineering

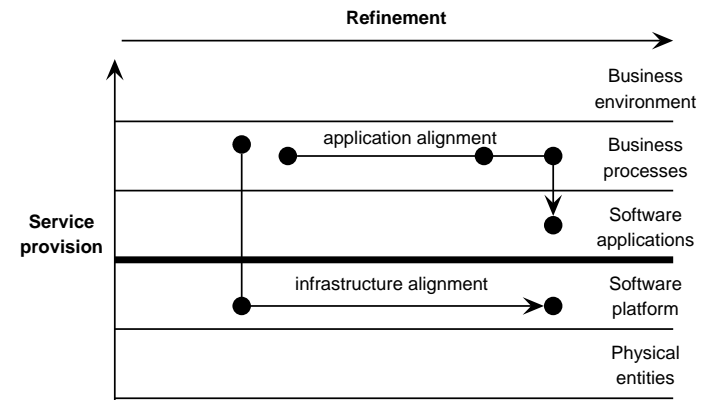
21

Reverse Conway (Hypothesis):

- A system too large to be maintained by one person, is maintained by a group of persons.
- The maintenance group of such a system will be isomorphic to the aspects (knowledge areas) of the system.
- Problems will cut across aspects.

22

4.3 Noncommutative alignment



See also [1].

23

4.4 Application alignment

- **Functional decomposition.** Encapsulate each service in one component.
- **Communication-oriented decomposition.**
 - *Device-oriented decomposition:* one software component per external device interfaced with. (Multi-channeling.)
 - *Actor-oriented decomposition:* One component per actor interfaced with.
 - *Event-oriented decomposition:* one software component per interesting subject domain event.
- **Behavior-oriented decomposition:** Encapsulate behavior to monitored or controlled in one component.
- **Entity-oriented decomposition:** one software component per interesting subject domain entity.

24

Open problems

- Subject area partitioning
- Process area partitioning
- Incorporating legacy
- Distributing ownership
- Using domain theories

25

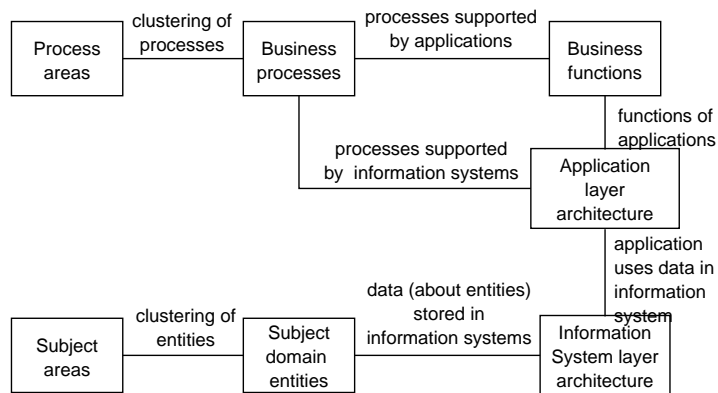
4.5 Infrastructure alignment

Four types of considerations

- Business goals
- Business problems
- Current systems
- Current technology trends

27

Method for application alignment



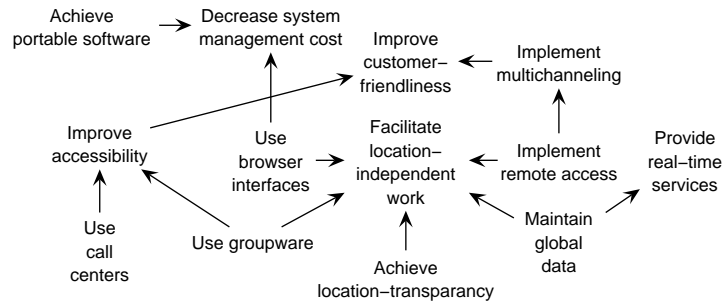
26

Observations

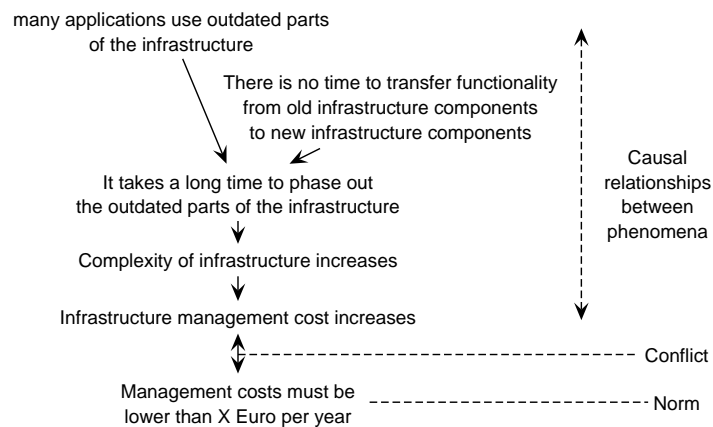
- Infrastructure domains (OS, Network, Middleware, DBMS, WFMS, ERP, Document management, Office, ...)
- Conway for infrastructure: domain architects organized by domain
- Domain islands: Highly specific technical knowledge. Distance between domains, with application, with business.
- Technology-driven decisions
- No traceability from infrastructure decisions to business goals and problems.
- Old technology never dies

28

Goal graph: rational reconstruction



Problem analysis: rational reconstruction



4.6 Coordination patterns

- Application architect to customer: Network
 - Asset specificity is customer-specificity
- Infrastructure architect
 - Asset specificity is domain-specificity
- Management to architect: Market
 - Result-driven
 - Project-driven versus architecture-constrained development
- Architects to architects: hierarchy
 - Instructions must be executed
 - Machine bureaucracy
- Support from management, users, builders, ...
 - Shared knowledge about business, IT
 - Shared norms: network coordination

5 Further work

- More case studies
- Make web-based handbook for application alignment
- Investigate infrastructure design arguments: goals, problems, quality, cost, ...
- Investigate coordination patterns

References

- [1] J.C. Henderson and N. Venkatraman. Strategic alignment: leveraging information technology for transforming operations. *IBM Systems Journal*, 32(1):4–16, 1993.
- [2] R.J. Wieringa. *Requirements Engineering: Frameworks for Understanding*. Wiley, 1996.
- [3] R.J. Wieringa. A survey of structured and object-oriented software specification methods and techniques. *ACM Computing Surveys*, 30(4):459–527, December 1998.
- [4] R.J. Wieringa. *Design Methods for Reactive Systems: Yourdon, Statestate and the UML*. Morgan Kaufmann, 2003.
- [5] R. J. Wieringa, H. M. Blanke, M. M.

ing application architecture to the business context. In *Conference on Advanced Information System Engineering (CAiSE 03)*, pages 209–225, 2003. LNCS 2681.