TECHNISCHE UNIVERSITÄT DRESDEN

DRESDEN concept

Ringvorlesung GRK „Role-based Software Infrastructures (RoSI)", funded by DFG
https://rosi-project.org

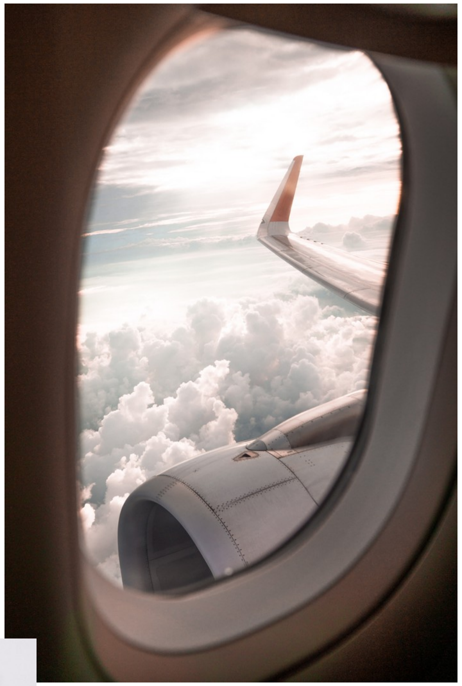
Development, Deployment, and Runtime of Context-Aware Software Systems

Uwe Aßmann

Version 0.4, WS 19/20 10/3/19

Welcome to RoSI!

- „ever-changing contexts“
  - Mobility
  - Personalization
  - Resource availablity
- How to realize
  - Adaptation to change of context?
  - Context polymorphism?

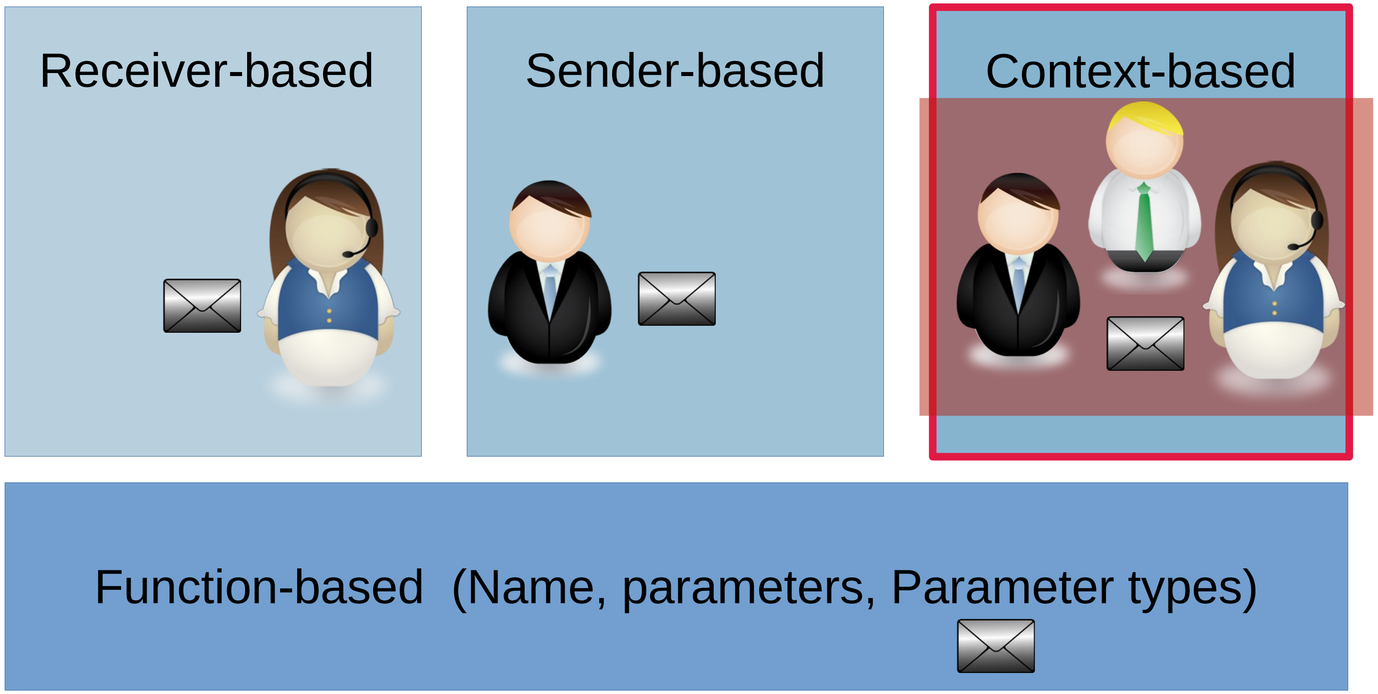


https://commons.wikimedia.org/wiki/File:2018-01-11_Olympiaeinkleidung_Deutschland_2018_by_Sandro_Halank%E2%80%9356.jpg

Photos by Alex Azabache on Unsplash

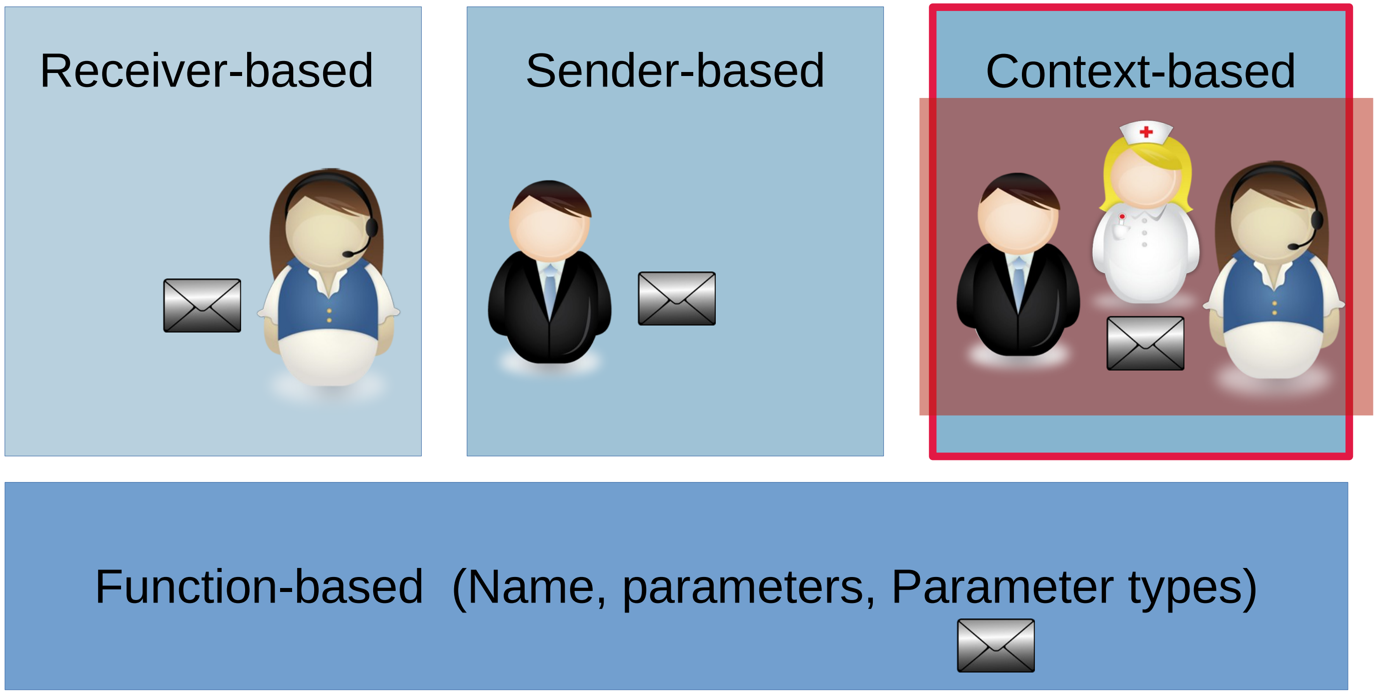
Multi-Dimensional Dispatch for Multi-Polymorphism

- How is the semantics of a function (method) determined?



Multi-Dimensional Dispatch for Multi-Polymorphism

- How is the semantics of a function (method) determined?



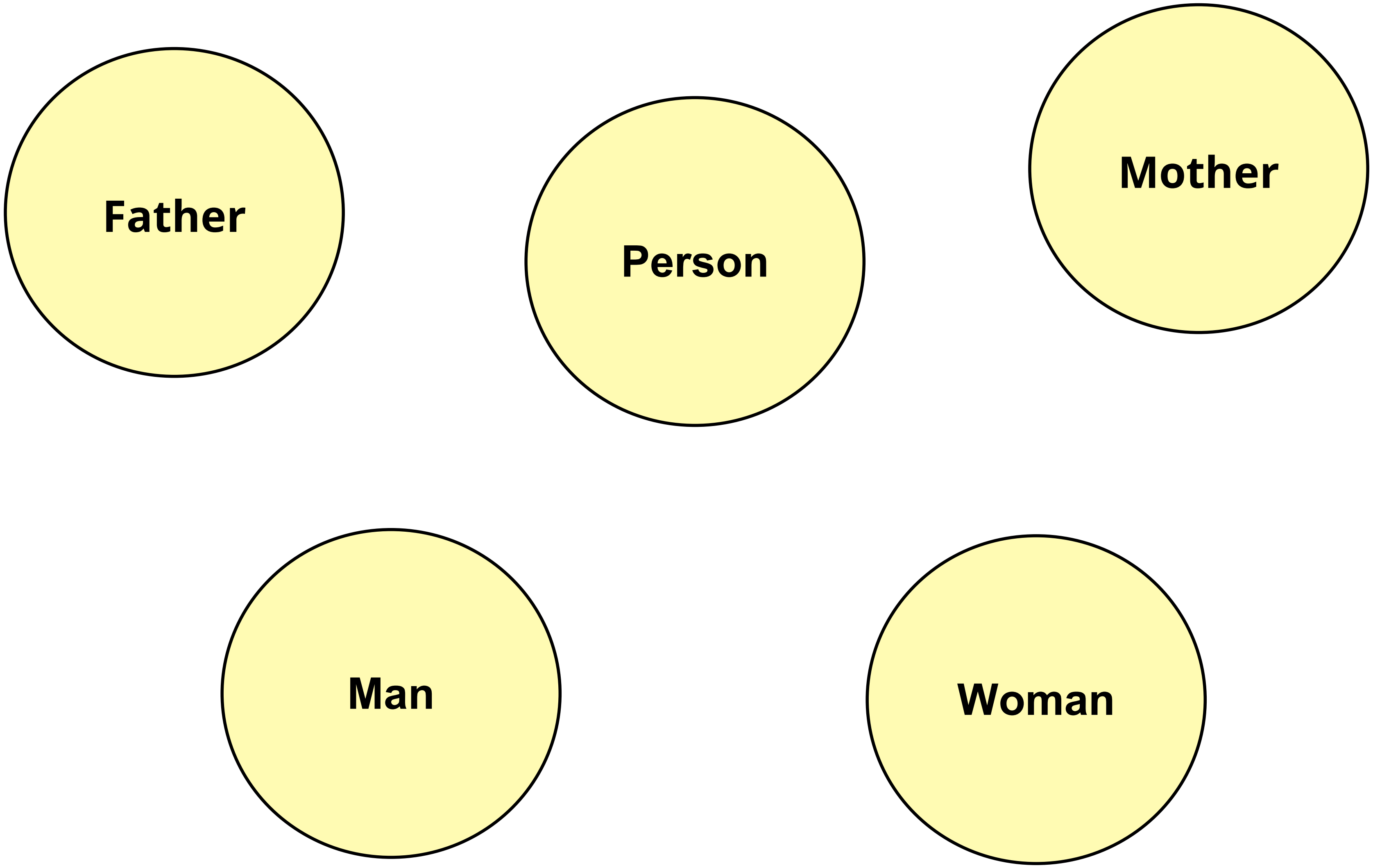
New Application Areas of ROSI

- Roles for context-sensitive cyber-physical systems (CPS)
  - Hypothesis: Role-contracts for safety and security
- Roles for emergence in Systems-of-Systems (SoS)
  - Hypothesis: Role models for unforeseen emergence

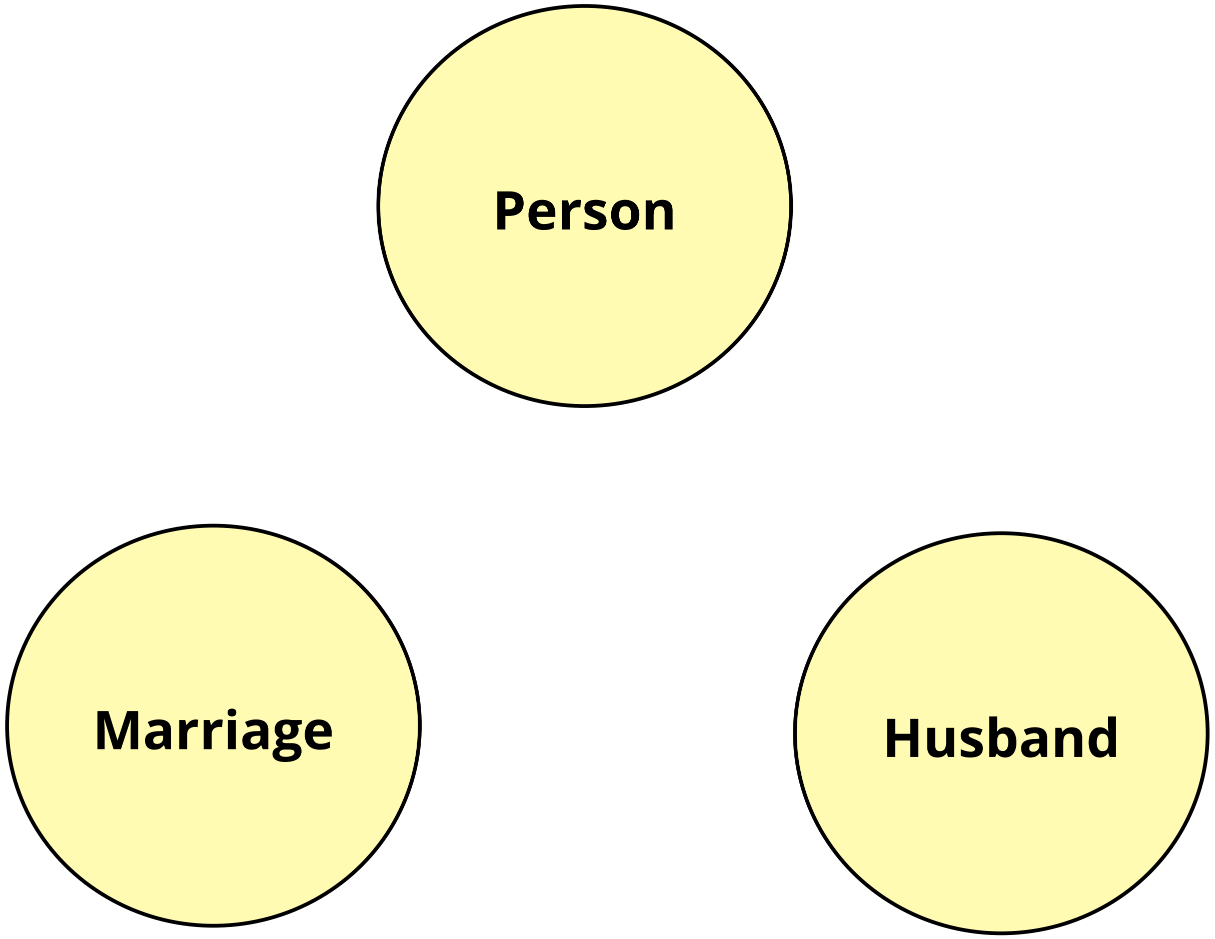
http://commons.wikimedia.org/wiki/File:Traffic_seen_from_top_of_Arc_de_Triomphe.JPG

http://commons.wikimedia.org/wiki/File:Jona_(SG)_-_L%C3%A4ttenhofweg_2011-04-08_14-54-48_ShiftN.jpg

A Riddle..



Another Riddle..



Role-Oriented Context-Aware Software Infrastructures (ROSI)

01. Contexts and Roles

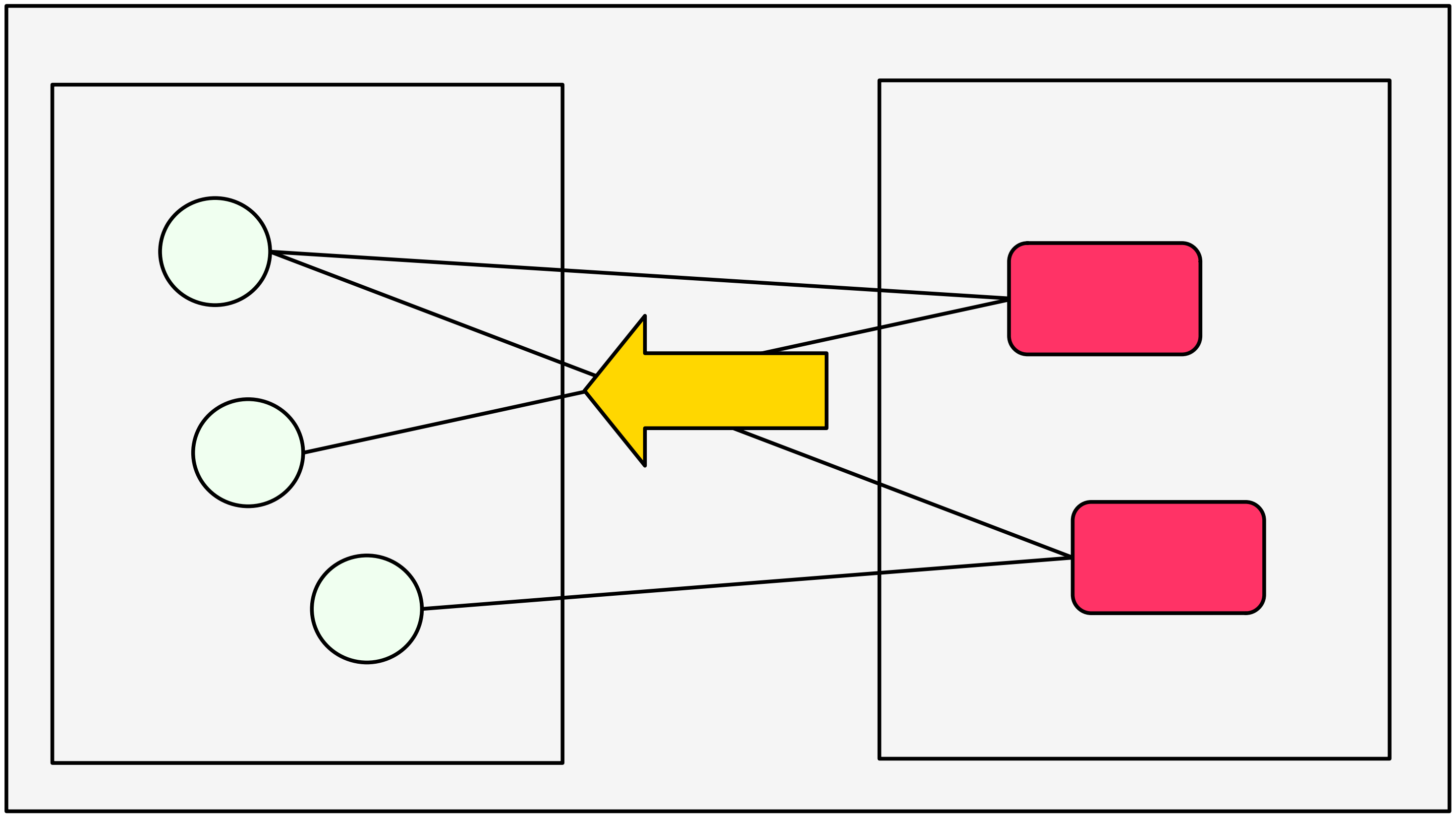
Overview of Talk

Role-Oriented Context-Aware Software Infrastructures (ROSI)

1.1. Adaptation Problems of the Standard Object Model

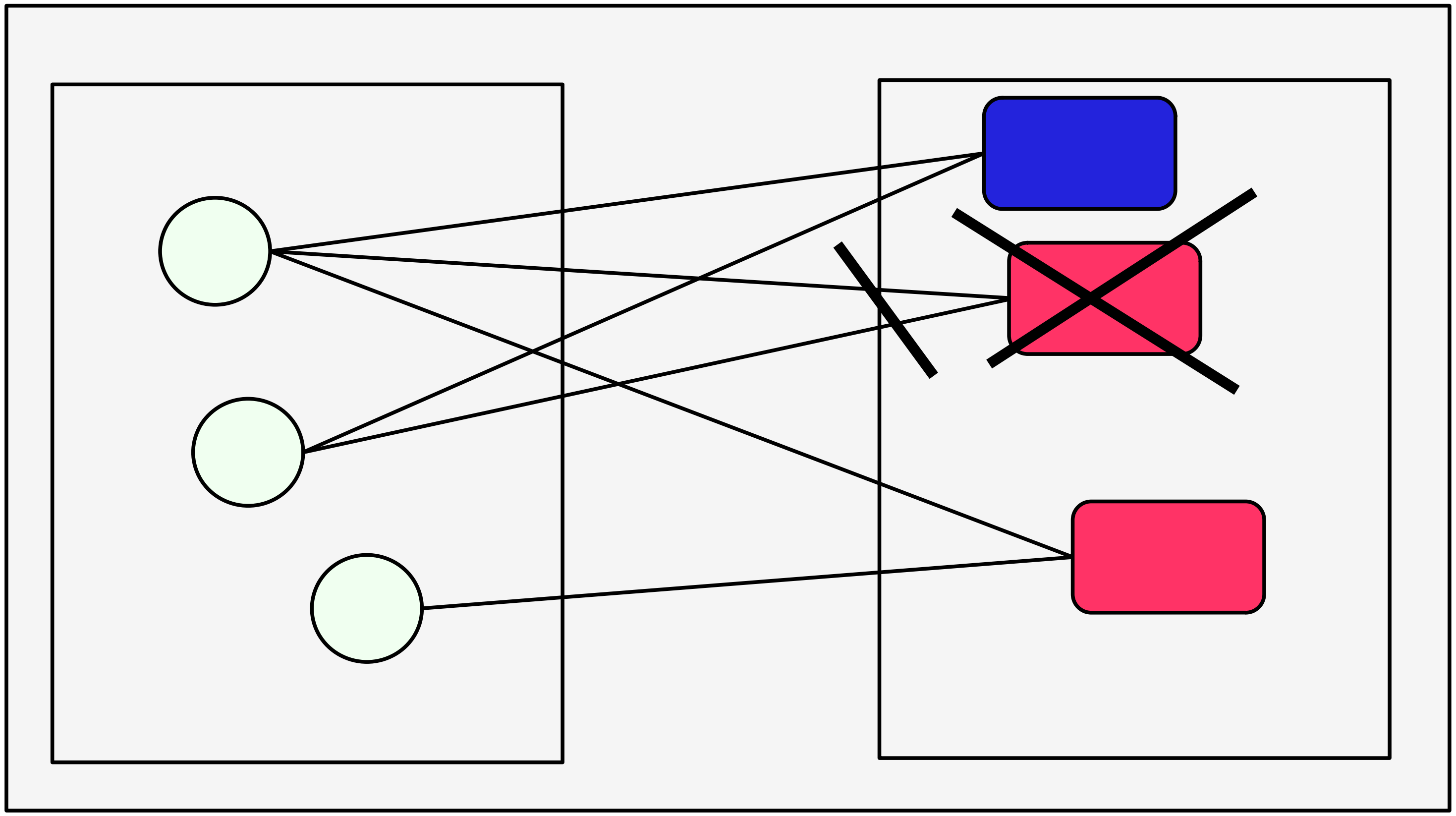
The Extensibility Problem

- How to extend software in a fine-grained way? Obliviousness?



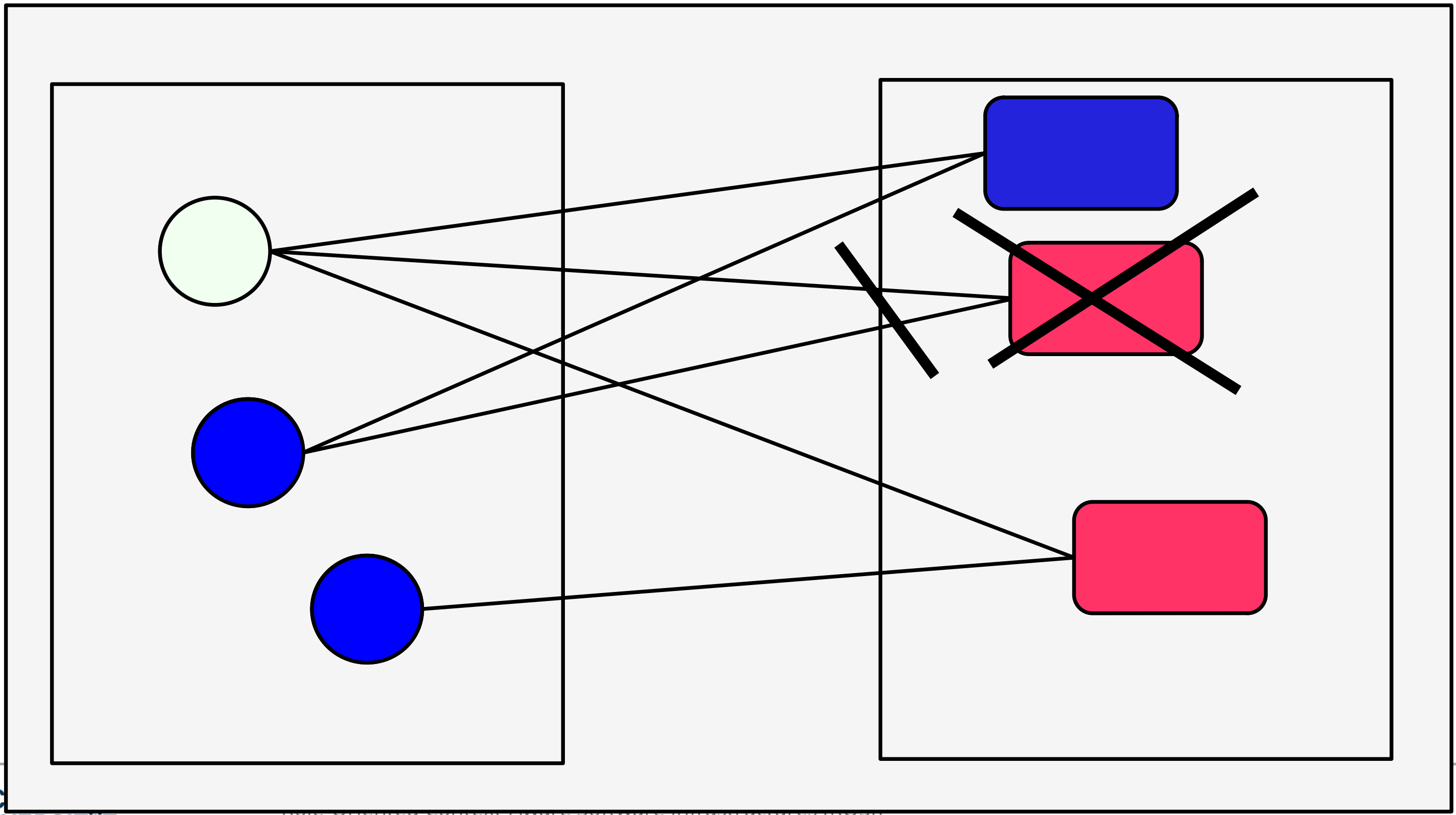
The Substitutability Problem

- How to substitute a component? (contracts necessary)



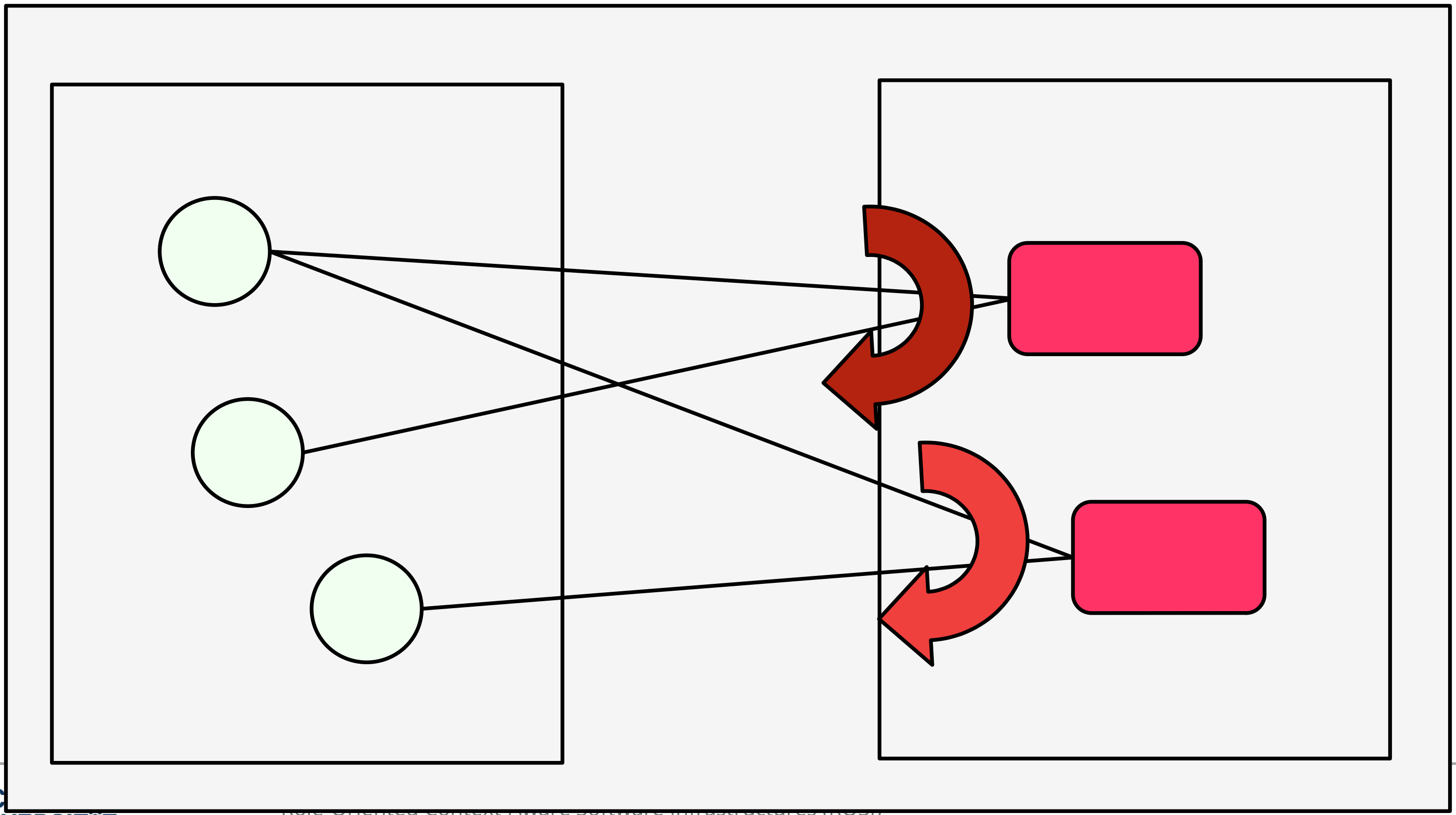
The Variability Problem

- How to vary many components, layers, slices, or a subsystem?



The Wrapping and Synchronization Problem

- How to wrap software with code, e.g., for protection or synchronization, transactions?



Big Problem 1) The Synchronization Problem (Inheritance Anomaly)

1980s: Parallel object-oriented languages (POOL, COOL)

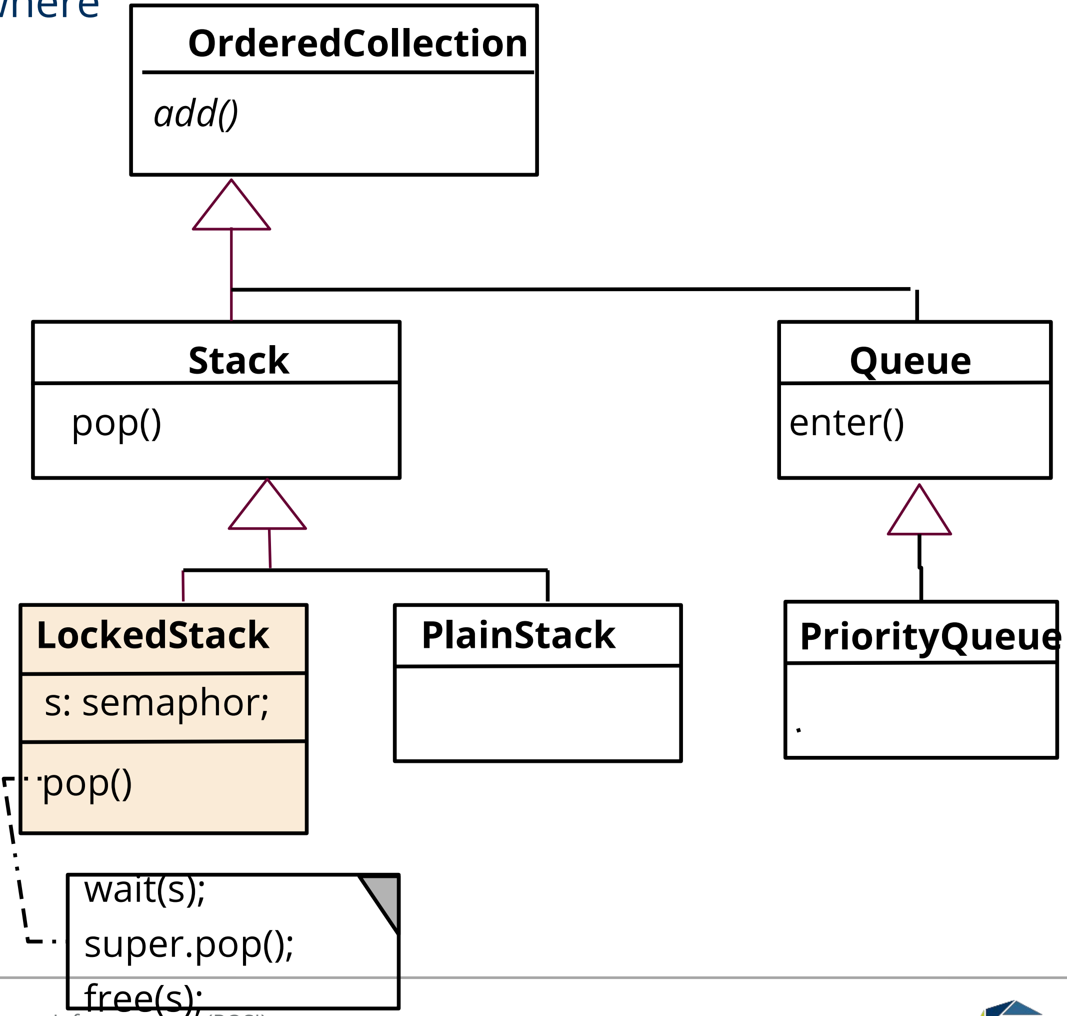
1991: Inheritance Anomaly

1988: Composition Technology (Aksit)
1994: Generic Synchronization Policies (McHale)
1988-94: Composition Filters (Aksit)

1996: Aspect-oriented Programming (AOP)
1996: D+COOL+RIDL (Lopes), 1999: Aspect/J (Kiczales)

Inheritance Anomaly - Why Concerns are Necessary

- In a parallel program or library, where should synchronization code be inserted?
  - Stack?
  - Queue?
  - OrderedCollection?
  - Collection?
  - Object?

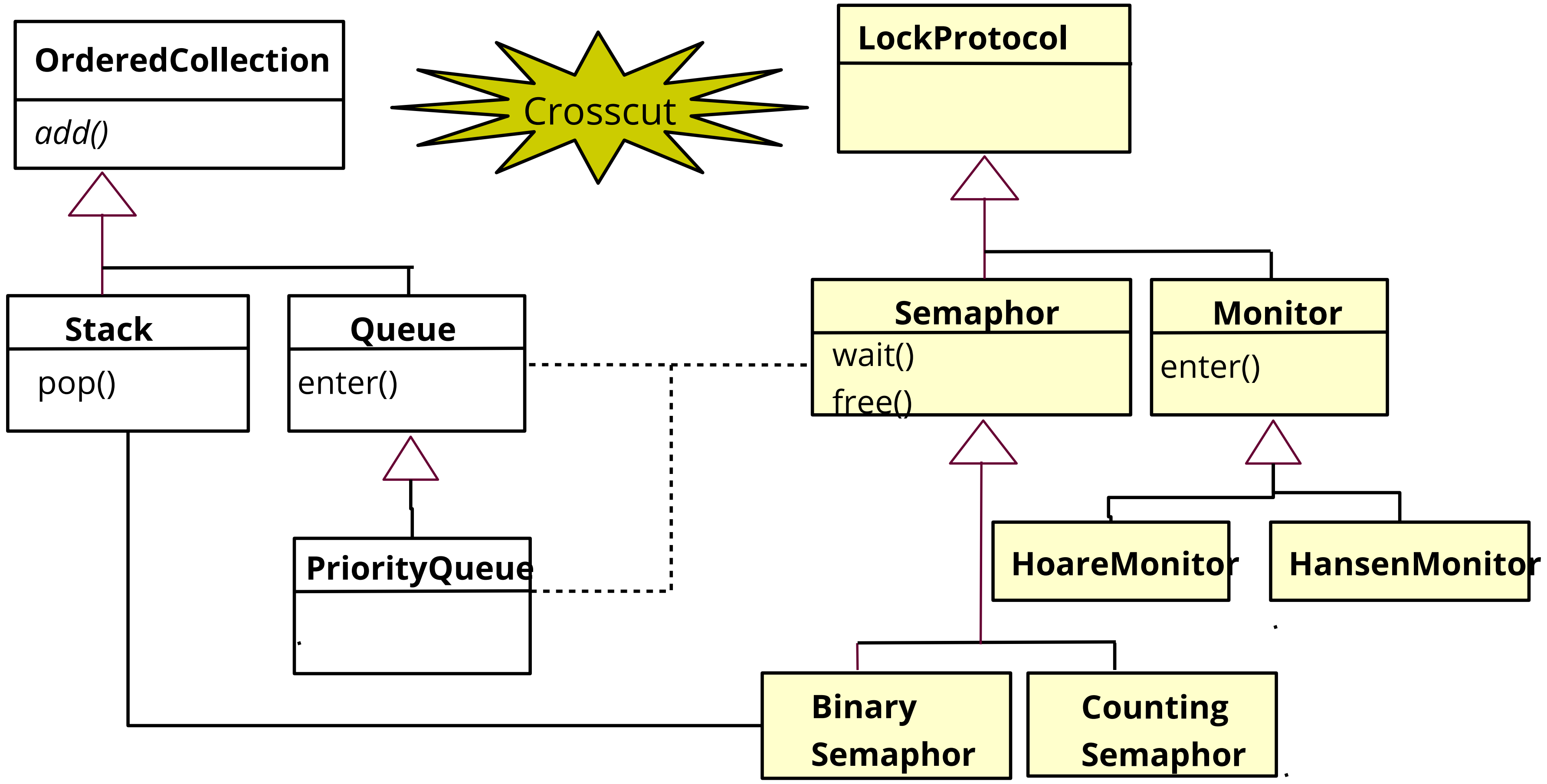


The Synchronization Problem (Inheritance Anomaly)

- At the beginning of the 90s, parallel object-oriented languages failed, due to the inheritance anomaly problem
  - *Inheritance anomaly*: In inheritance hierarchies, synchronization code is intermingled with the algorithm and cannot be easily exchanged
  - *Synchronization tangling*: Because synchronization code *braces* code, it is *tangling*
  - *Synchronization crosscut*: Because synchronization code *is reused* code, it is *crosscutting*

Algorithm and Synchronization are Two Different Concerns (Core and Aspect)

- Composition fixes crosscut between core and aspect

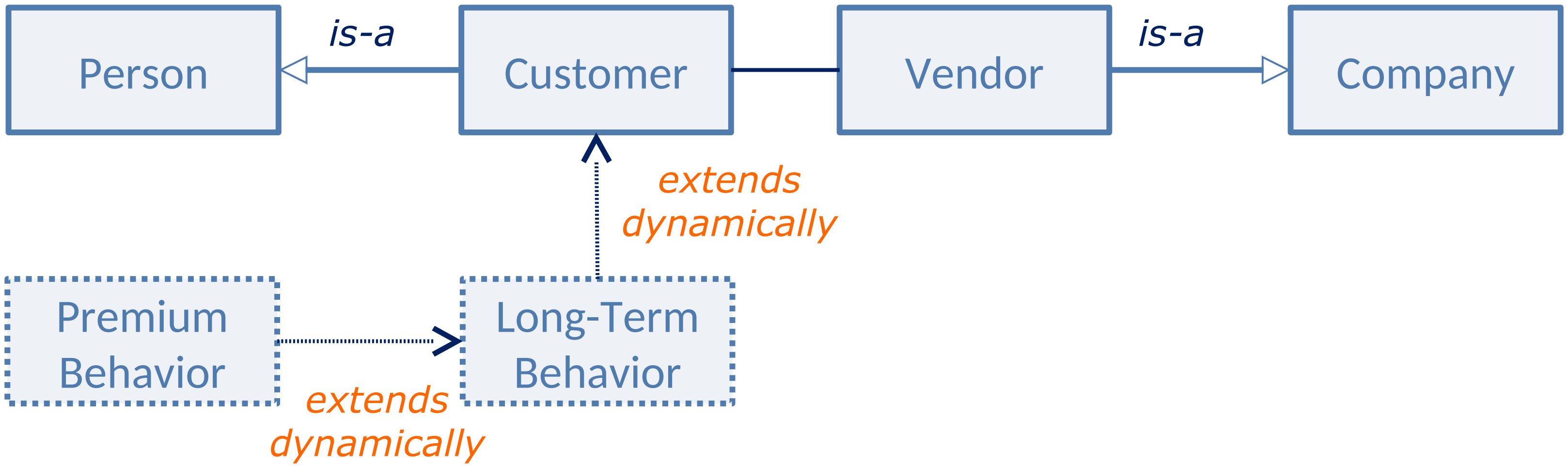


Problems of Aspect-Orientation

- Not well integrated into the standard OO model
- Semantics unclear
- Often only static

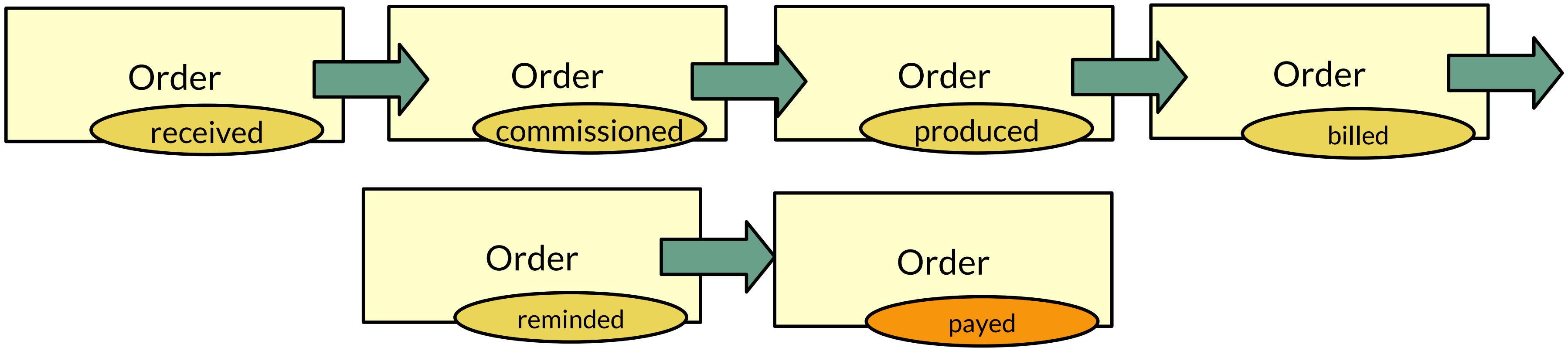
Big Problem 2) Run-time Adaptability
Negative Example: “San Francisco”-Framework of IBM

- Enterprise Resource Planning (ERP) in Java, 1995-99
- Dynamic extensions of classes and life-cycle automata
- Classic object-orientation too inflexible
- FAILED



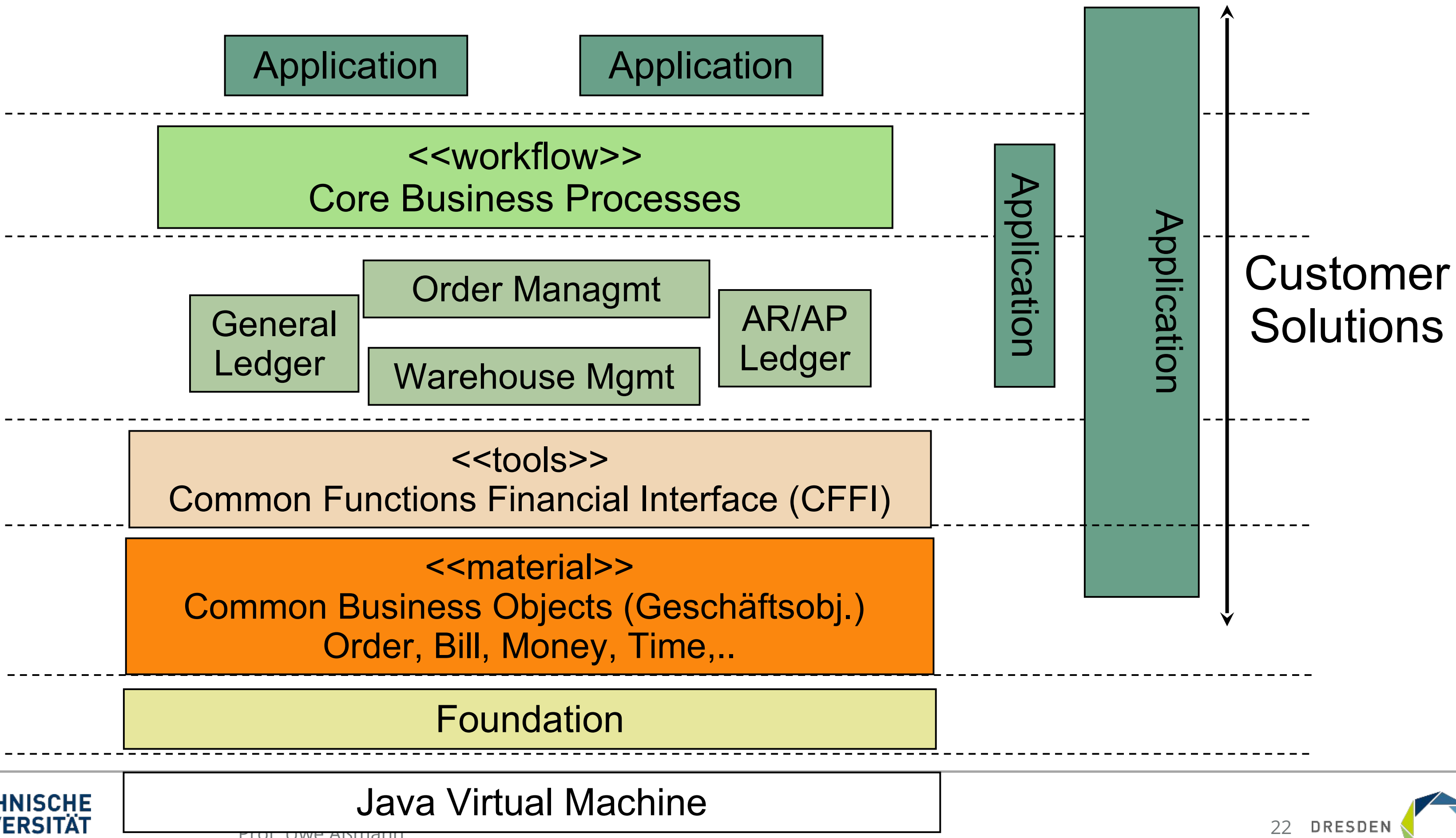
Business Objects

- In large ERP frameworks (see SAP) business objects get very complex
- Ex.: **Order**
  - Many phases and collaborators
  - Many states and roles
- Dynamic Extensibility and Variability (Adaptation) required

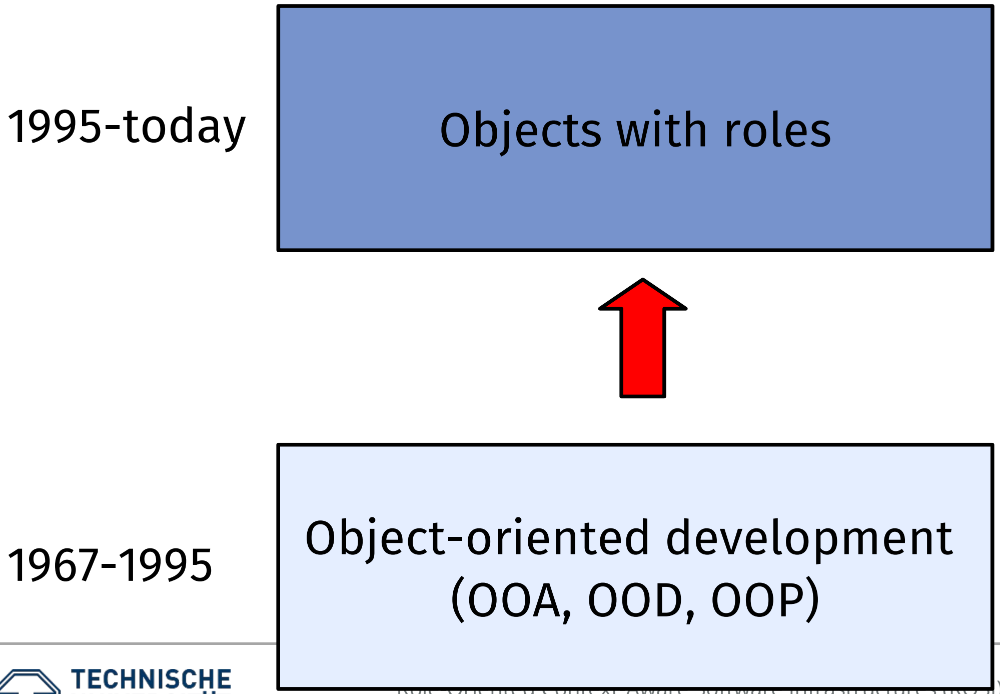


Architecture of IBM San Francisco ERP Java-Framework

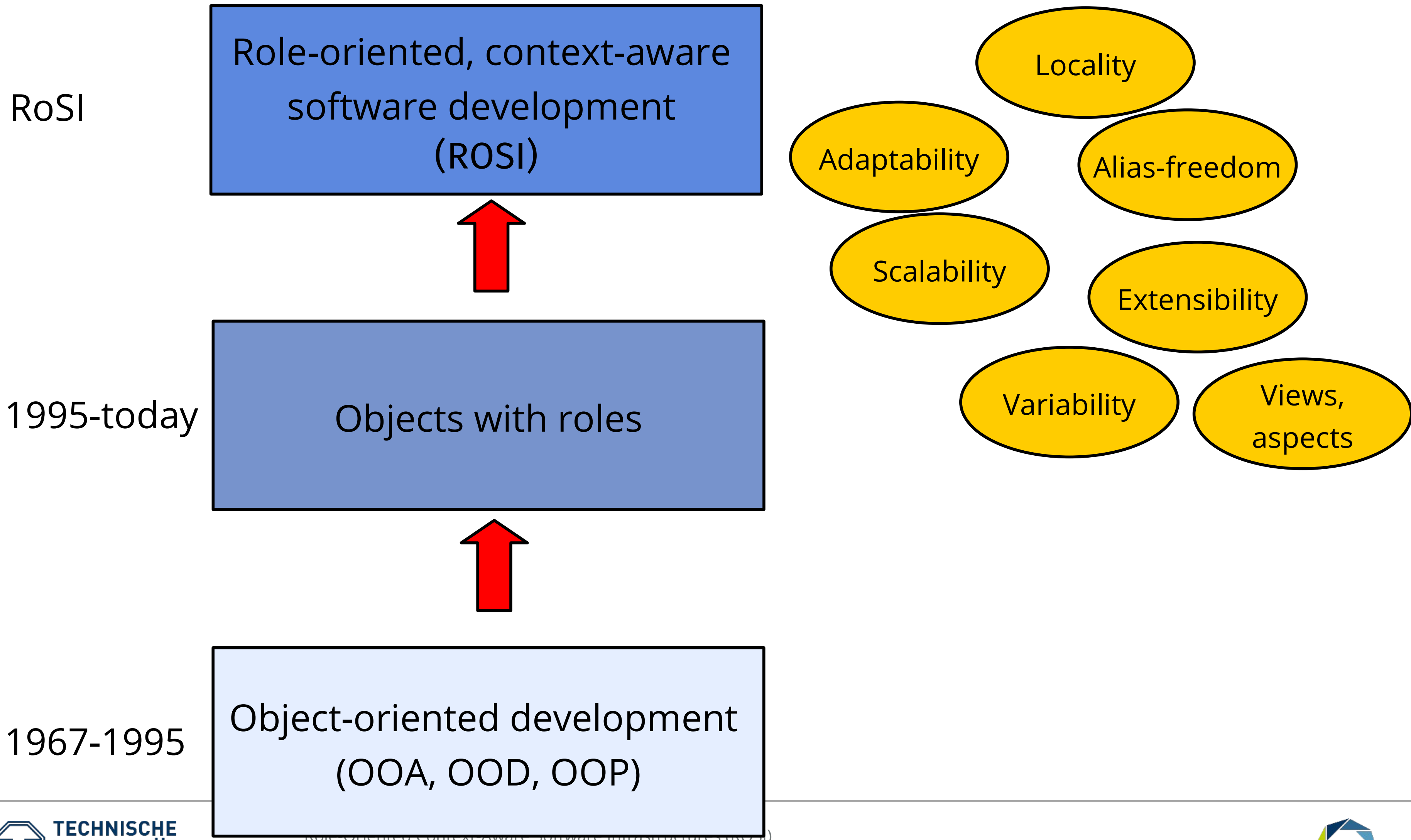
- P. Monday, J. Carey, M. Dangler. SanFrancisco Component Framework: an introduction. Addison-Wesley, 2000.



Ladder of Technologies

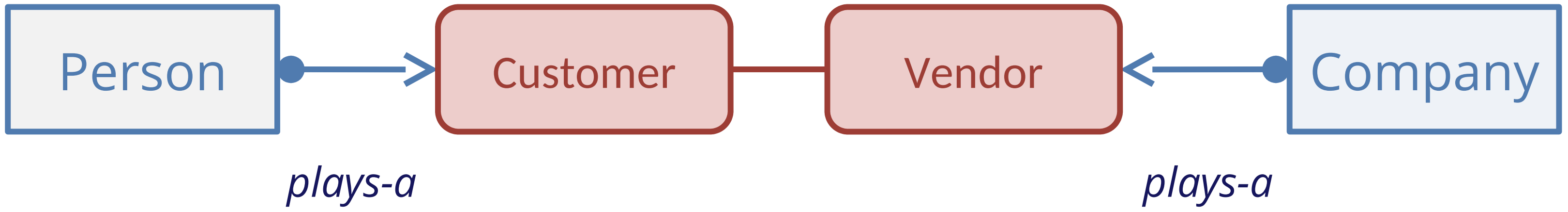


Ladder of Technologies



Role Modelling – a Hope

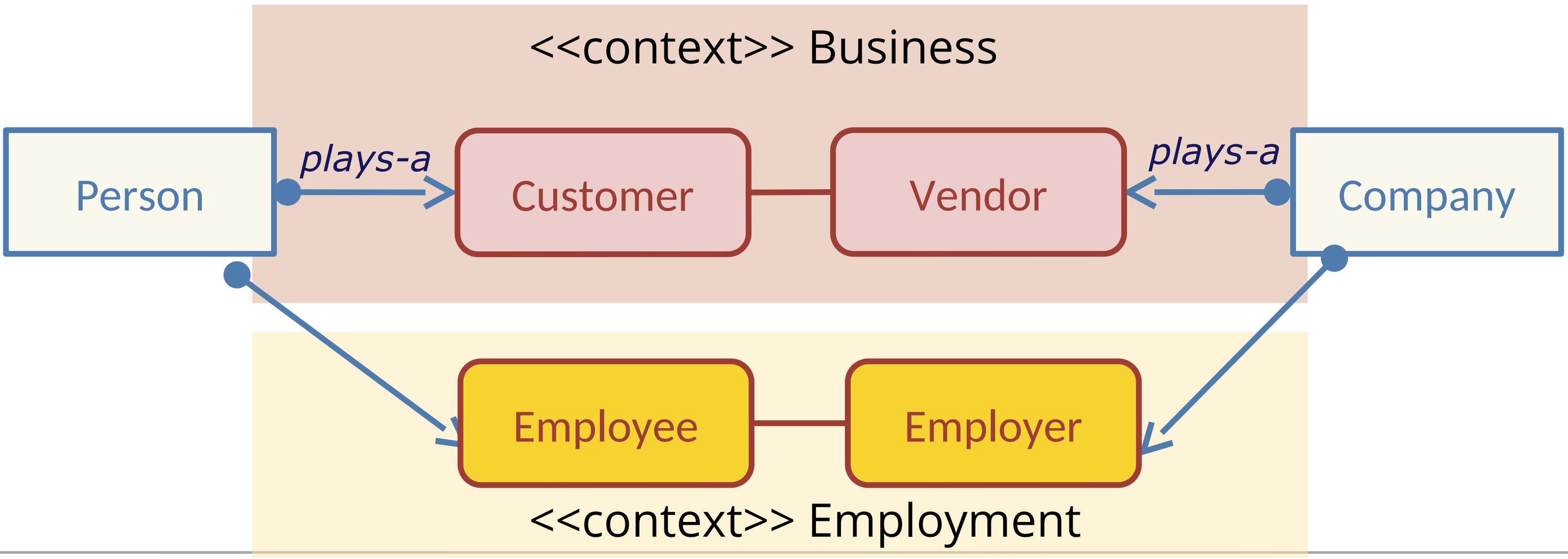
- Separate the **functional core** of an object of its **context-based a**nd **fluid** features



- Restrictions so far:
  - only used in singular fields of Computer Science
  - no cross-layer correspondance
  - no formalization

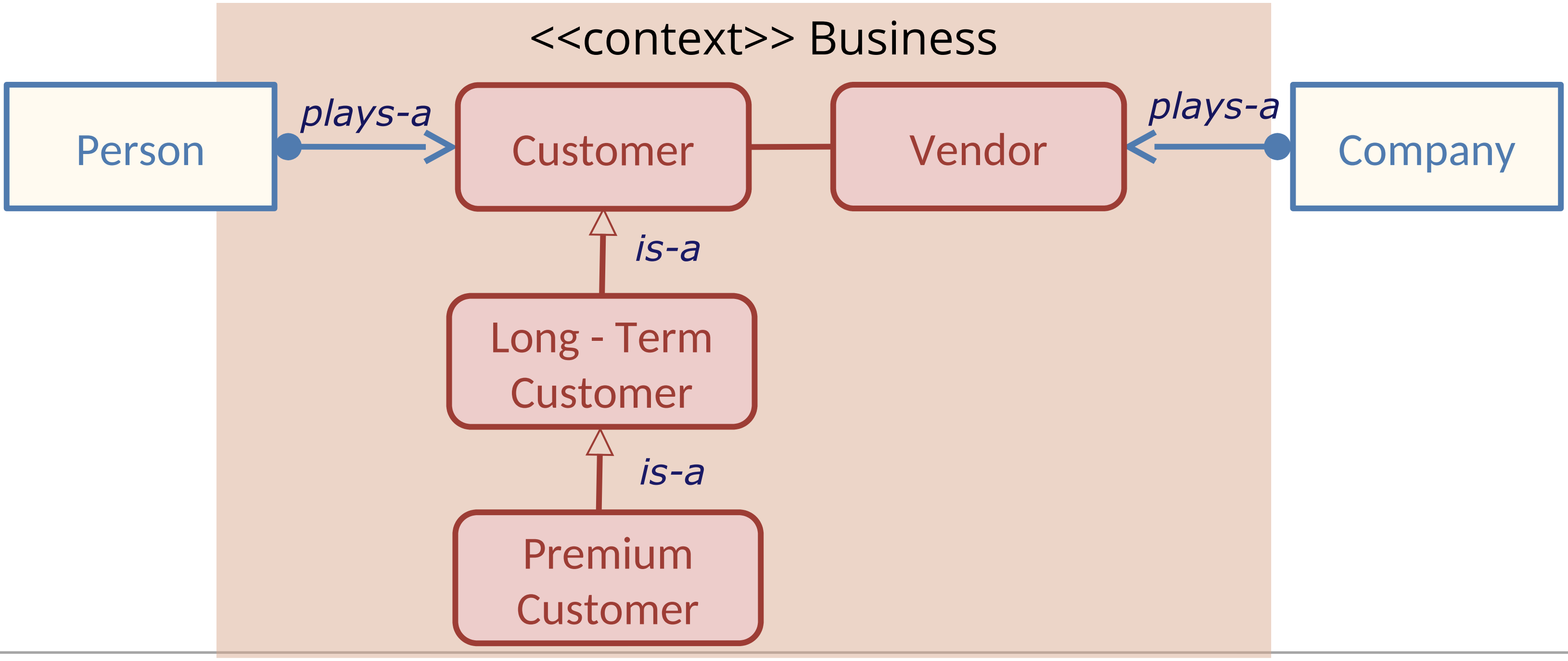
Example: Business Objects

- Extend behavior dynamically by **roles (context-based and fluid types)**
- Convention: Context is expressed by *background boxes* or *color*



Example: Business Objects

- Extend behavior dynamically by **roles (context-based and fluid types)**
- Refinement by role inheritance
-



The Hypothesis of Role-Oriented Context-Aware Development

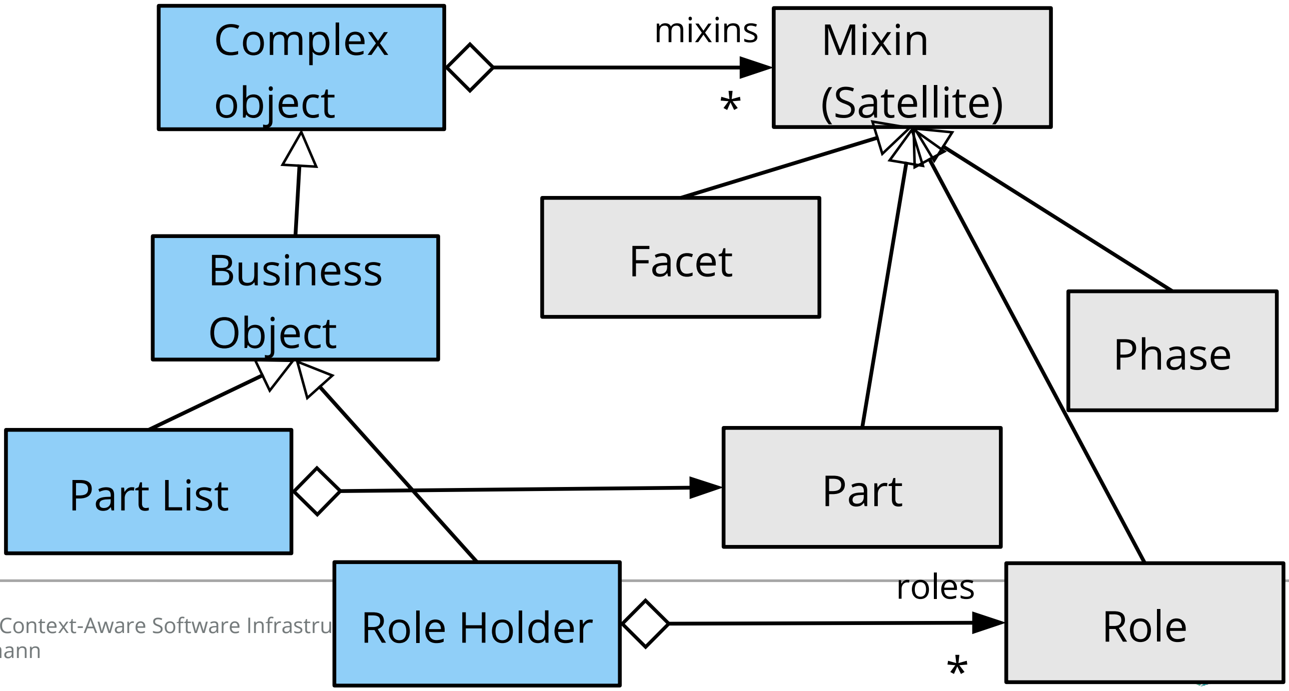
- ...is that **context-based features of objects and systems** can be modeled **with roles, cross-cutting**
  - all phases of the life-cycle
    - requirements, design, implementation, runtime
  - all levels of development
    - Concept modelling in metalanguages,
    - Language modelling,
    - Application modelling and programming,
    - Run-time
- and that this technology is **practically applicable.**

Role-Oriented Context-Aware Software Infrastructures (ROSI)

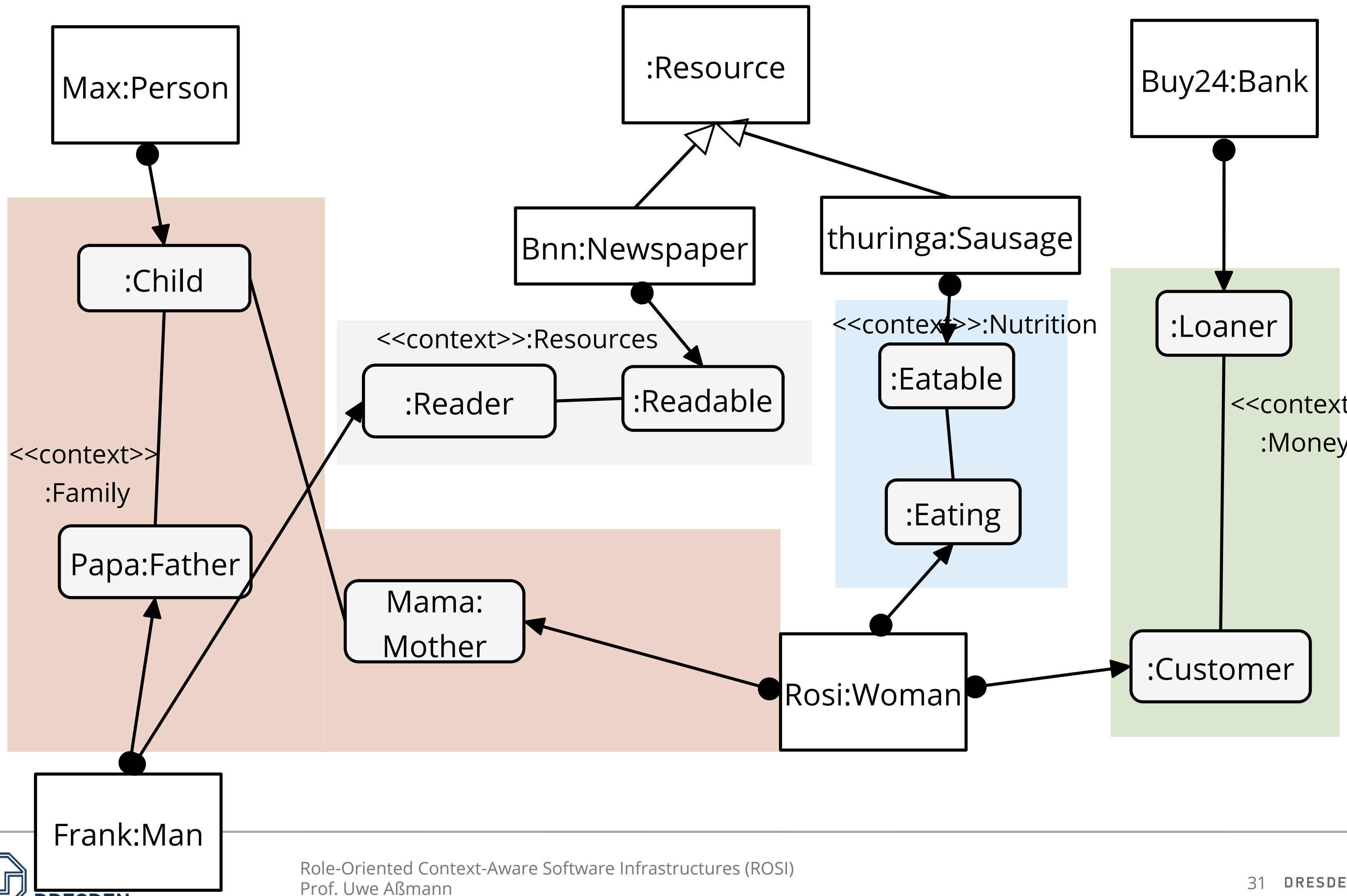
1.1.2. Scenario Families and Banks

Complex Objects

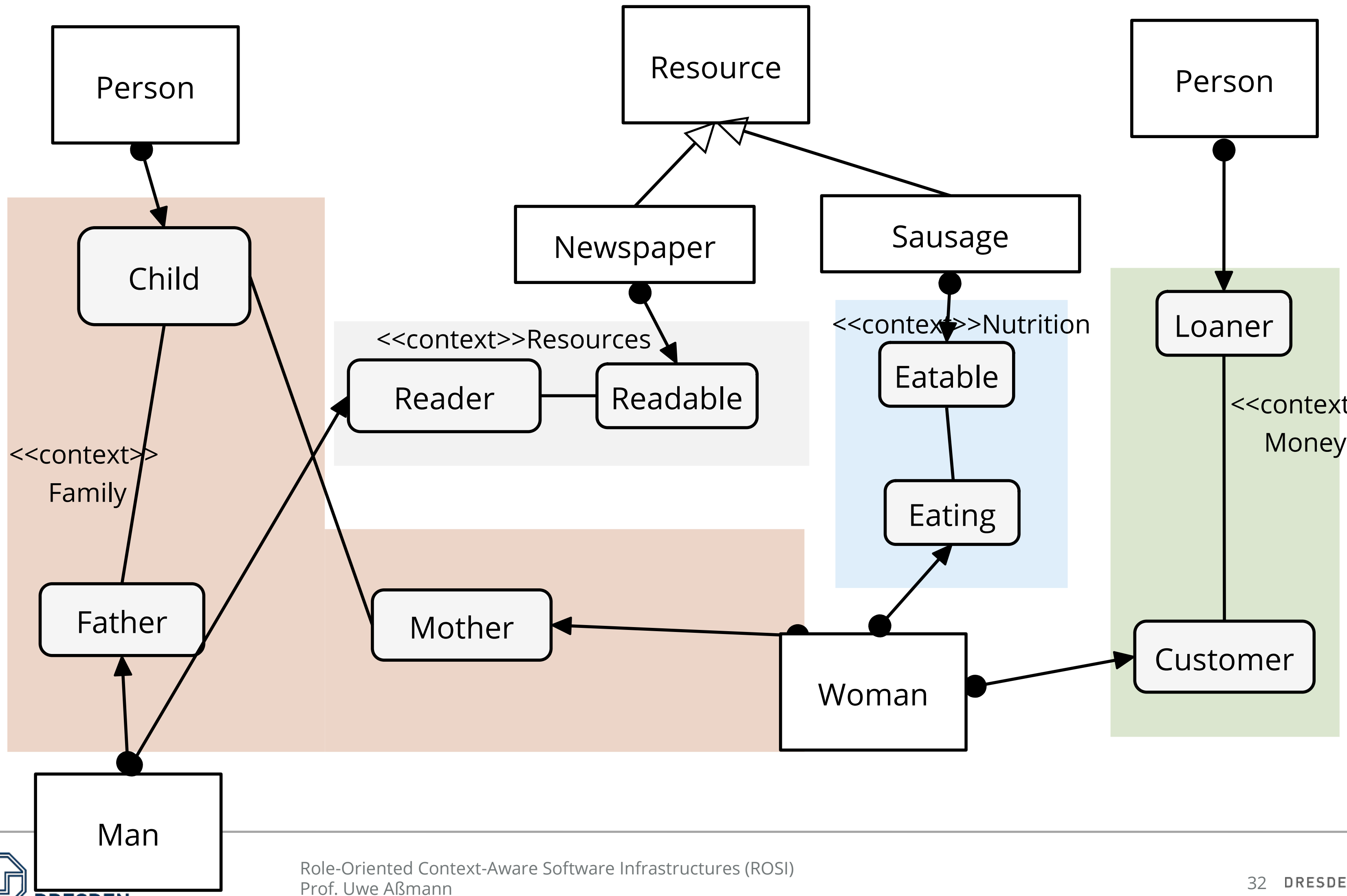
A **complex object (subject, compound object)** is a (logically coherent) object, represented in modeling and programming level by one ***Core*** and several ***Subobjects (mixins)***



Families, Resources and Banks (Snapshot, Object-Role Model)



Families and Banks in Natural and Role Types



Role-Oriented Context-Aware Software Infrastructures (ROSI)

1.2. Beyond Objects - Role Modeling and the Steimann Factorization of Types

Splitting a type into a tuple of natural and founded parts

Roles in the Literature

- Databases (Bachmann 77)
- ER model (Chen 76); though hidden in association ends
- OO Modeling (Reenskaug 95)
  - Design patterns (Riehle 98)
  - Course “Design patterns and frameworks” at TUD
- Product line engineering (Smaragdakis,Batory 02)
- Connectors in architectural languages (Garlan, Shaw 95)
- Security: Role-based Access Control (RBAC)
  - ACL lists in operating systems
- Ontologies (Brachman, description logic)
- ... [Steimann DKE 2000] has many more and tries to unify them
- UML has “collaborations” using role types
- [Kühn 2014] defines compartments as structured context objects

Rigid and Founded Types

If an object that has a ***rigid*** type, it cannot stop being of the type without loosing its identity [Guarino]

- Example:
  - *Book* is a rigid type, *Reader* is a non-rigid type
    - Reader can stop reading, but Book stays Book
- Rigid types are *tied to the identity* of objects
  - A *non-rigid type* is a dynamic type that is indicating a state of the object

A ***founded type*** *(relative type)* is a type that exists always in collaboration (association) with another class.

Role and Natural Types

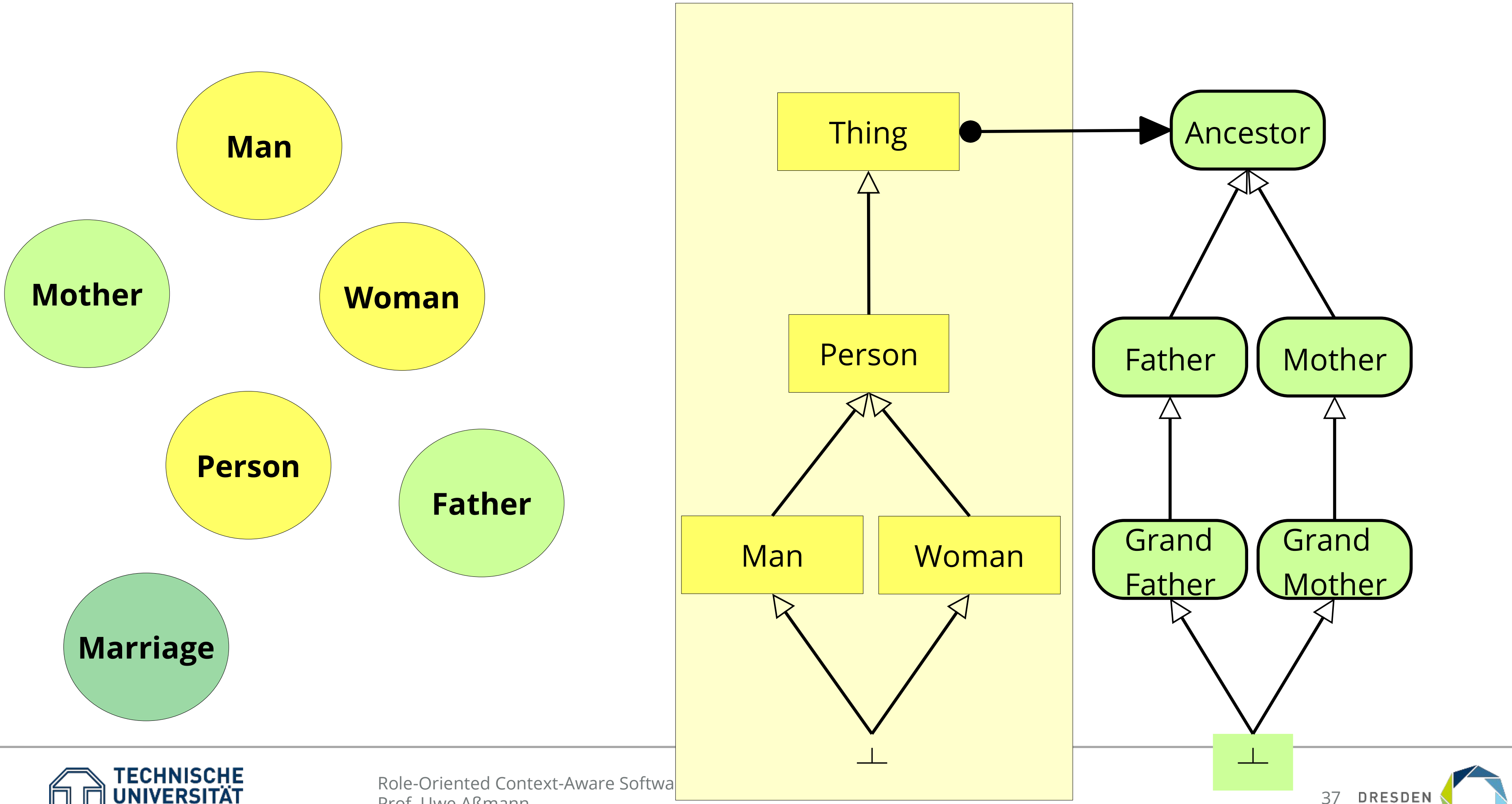
A ***role type*** is a founded and non-rigid type.

Role types are in collaboration and if the object does no longer play the role type, it does not give up identity.

A ***natural type*** *is* non-founded and rigid.

A natural type is *independent* of a relationship.
The objects cannot leave it.

Solution to the Little Riddles..



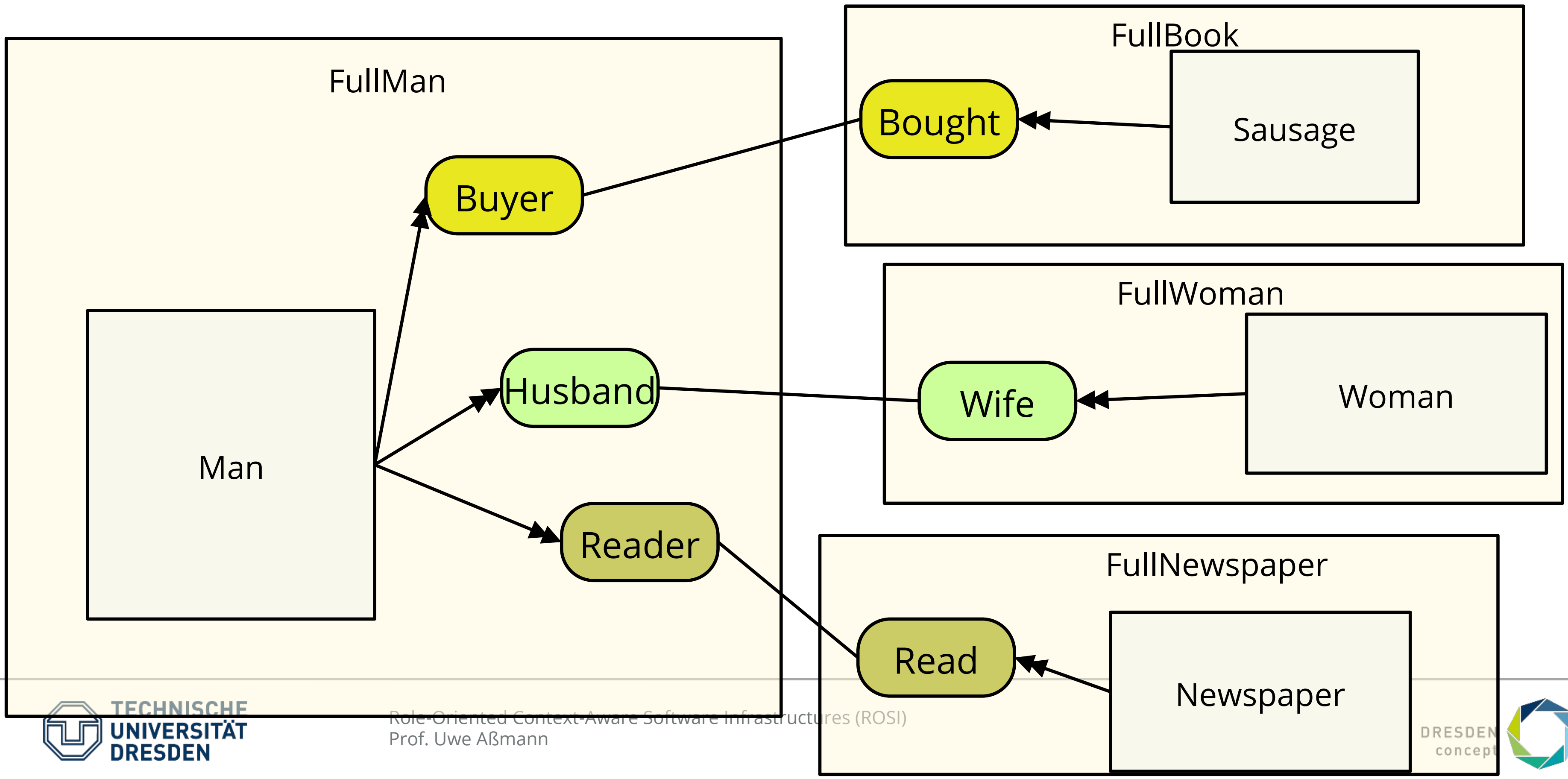
Role Types are Metatypes

- A **metatype** describes a type (is a type of a type)
  - Rigid Type
  - Natural Type
  - Founded Type
  - Role Type

Hypothesis:
The distinction of metatypes promotes
Separations of Concerns.

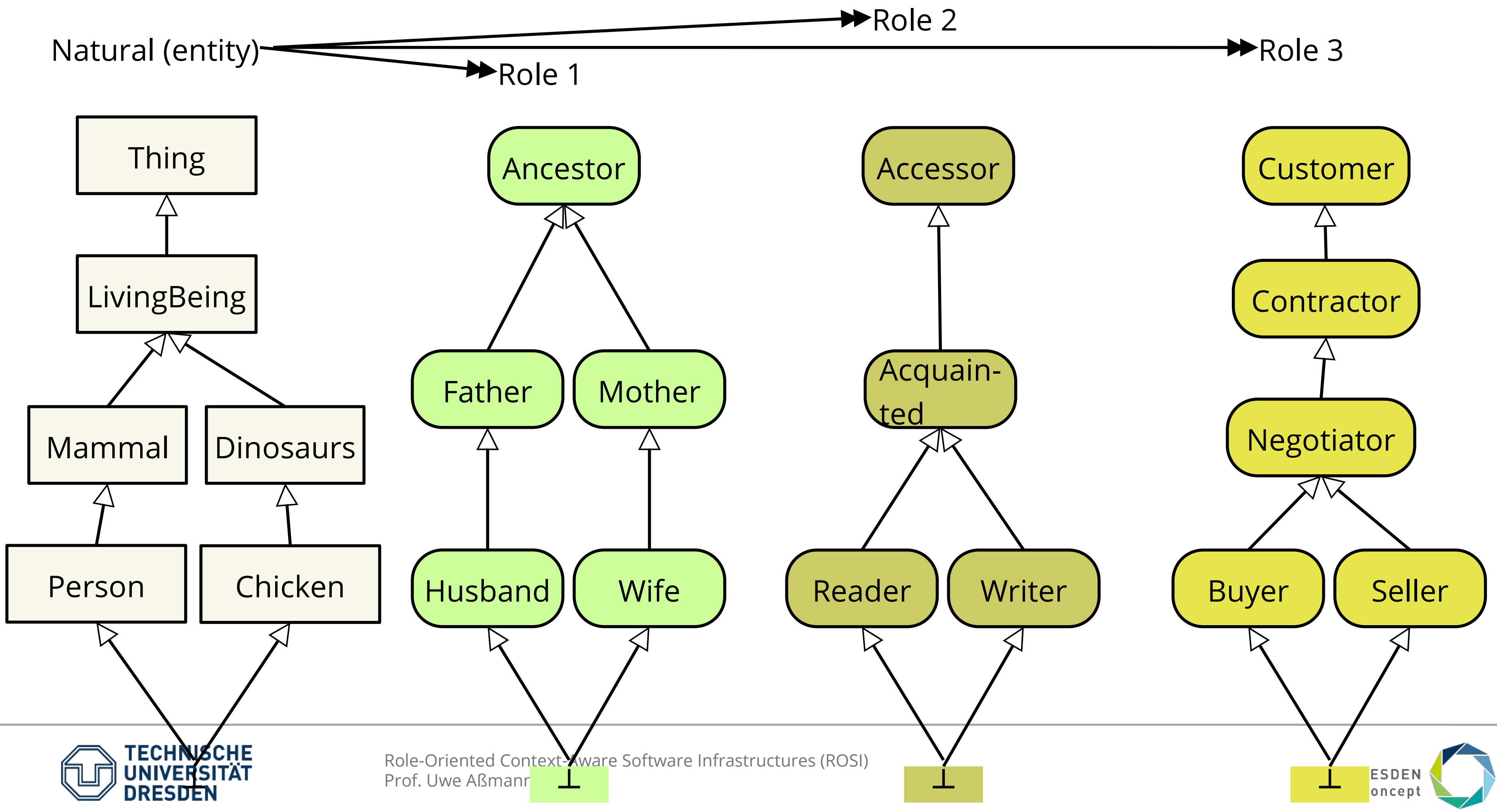
Steimann Factorization [Steimann, DKE 2000]

- Splitting a full type into its *natural* and *role-type* components
  - FullType = Natural x (role-type, role-type, ...)
  - FullMan = Man x (Reader, Husband, Customer, ..)



Full Type is from an Inheritance Product Lattice

- What is a reading buying husband person?



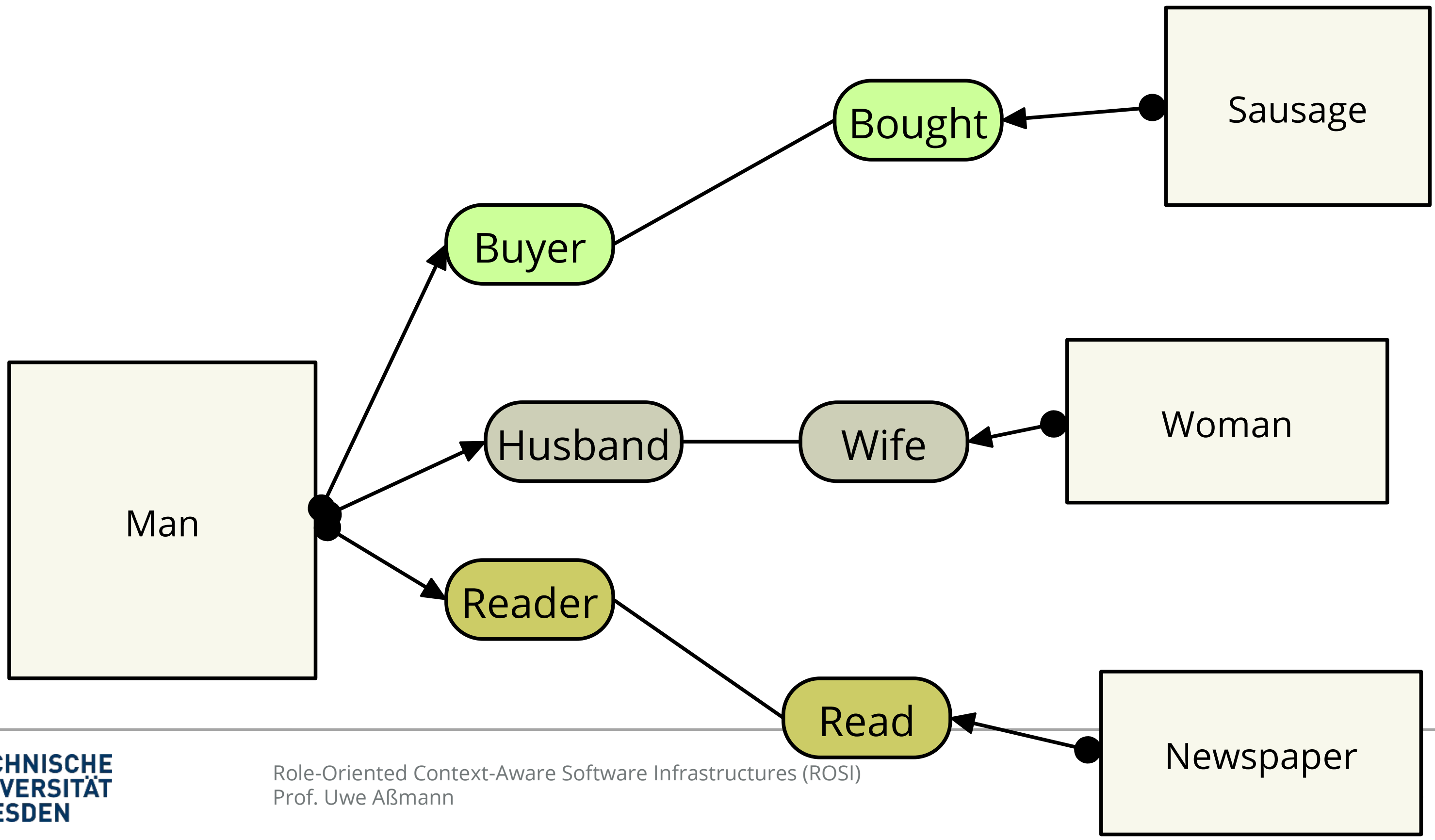
The Steimann Factorization

- Simpler, multi-dimensional inheritance hierarchies (product lattice)

Divide (partition) a *type* into a *tuple type* over a product lattice of a core dimension and n-1 role dimensions(Core, Role_1, ..., Role_n)

Concern-Separated Representation of Object Nets

- Collaborations (Role models) are interprocedural slices and belong to contexts
- Collaboration schemas are schemas for interprocedural slices



ROSI – Role-Oriented Context-Aware Software Infrastructures

1.2.1 Contexts and Compartments

[Kühn 2014]

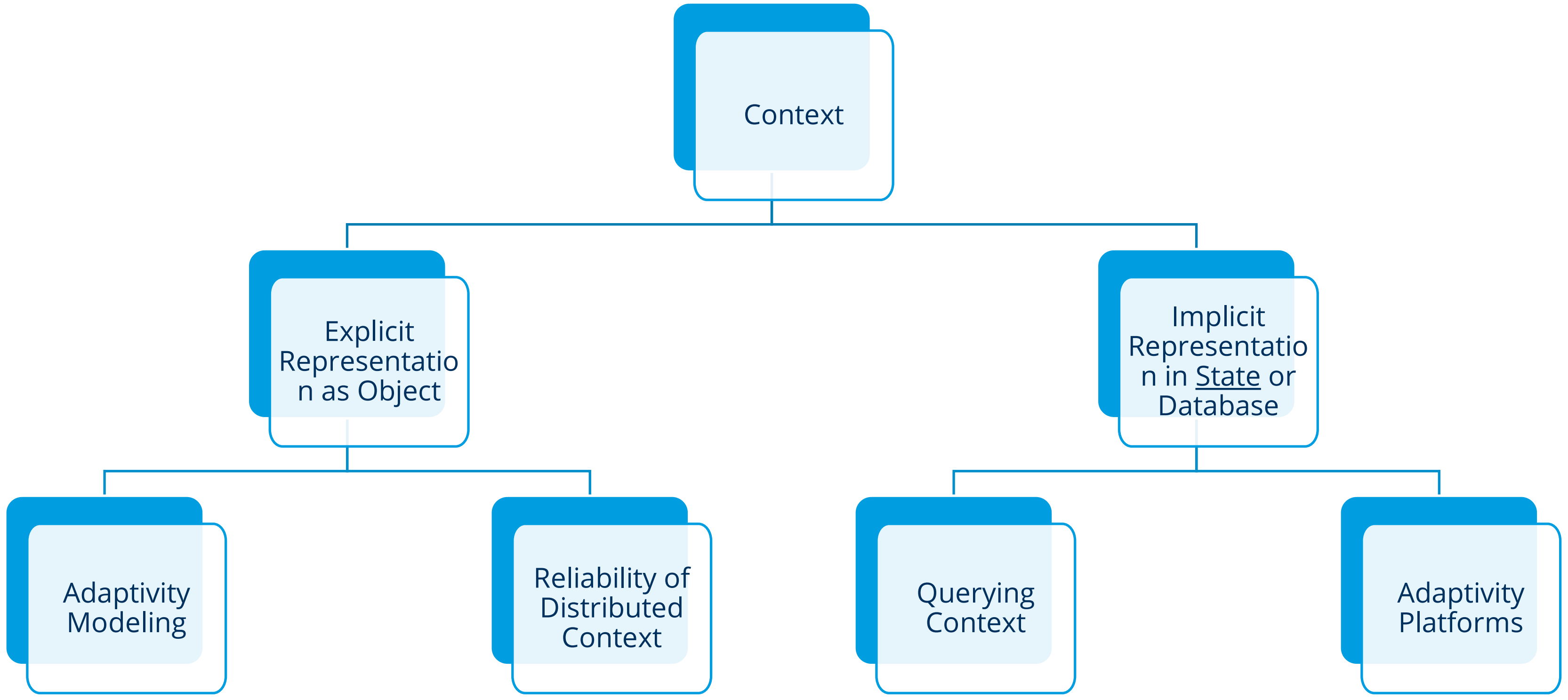
How to Model Contexts

- A ***context*** is an object reifying contextual conditions, *activating* and *deactivating* a set of roles of a set of objects
  - Contexts show that contextual conditions hold
  - *Marriage* (enables Husband and Wife)
  - *Light* (enables reading)
- A ***compartment*** is a structured context *activating* and *deactivating subcontexts*
  - *Marriage*: Mistress (Mätresse) enables lover and lovee during Marriage
  - *Light*: Glasses (enables reading while light is on)
- A ***compartment hierarchy*** is a hierarchy of structured contexts
  - *World model* (town, building, room)
- A ***compartment forest*** is a multi-hierarchy of structured contexts
  - *World model and company model*

Photo by ROOM on Unsplash

Photo by Alex Azabache on Unsplash

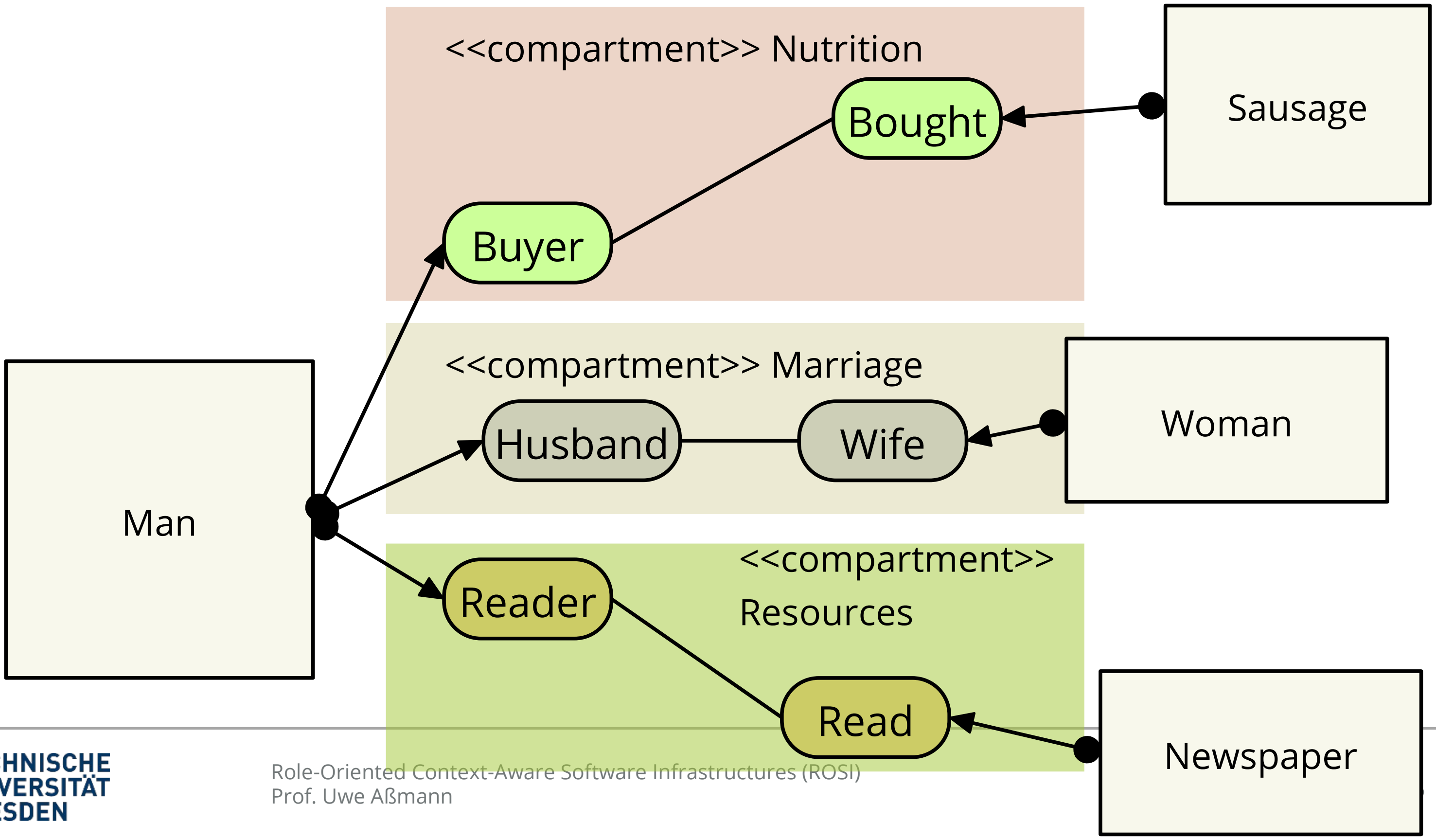
Explicit and Implicit Contexts


Context
Explicit Representation as Object
Implicit Representation in State or Database
Adaptivity Modeling
Reliability of Distributed Context
Querying Context
Adaptivity Platforms

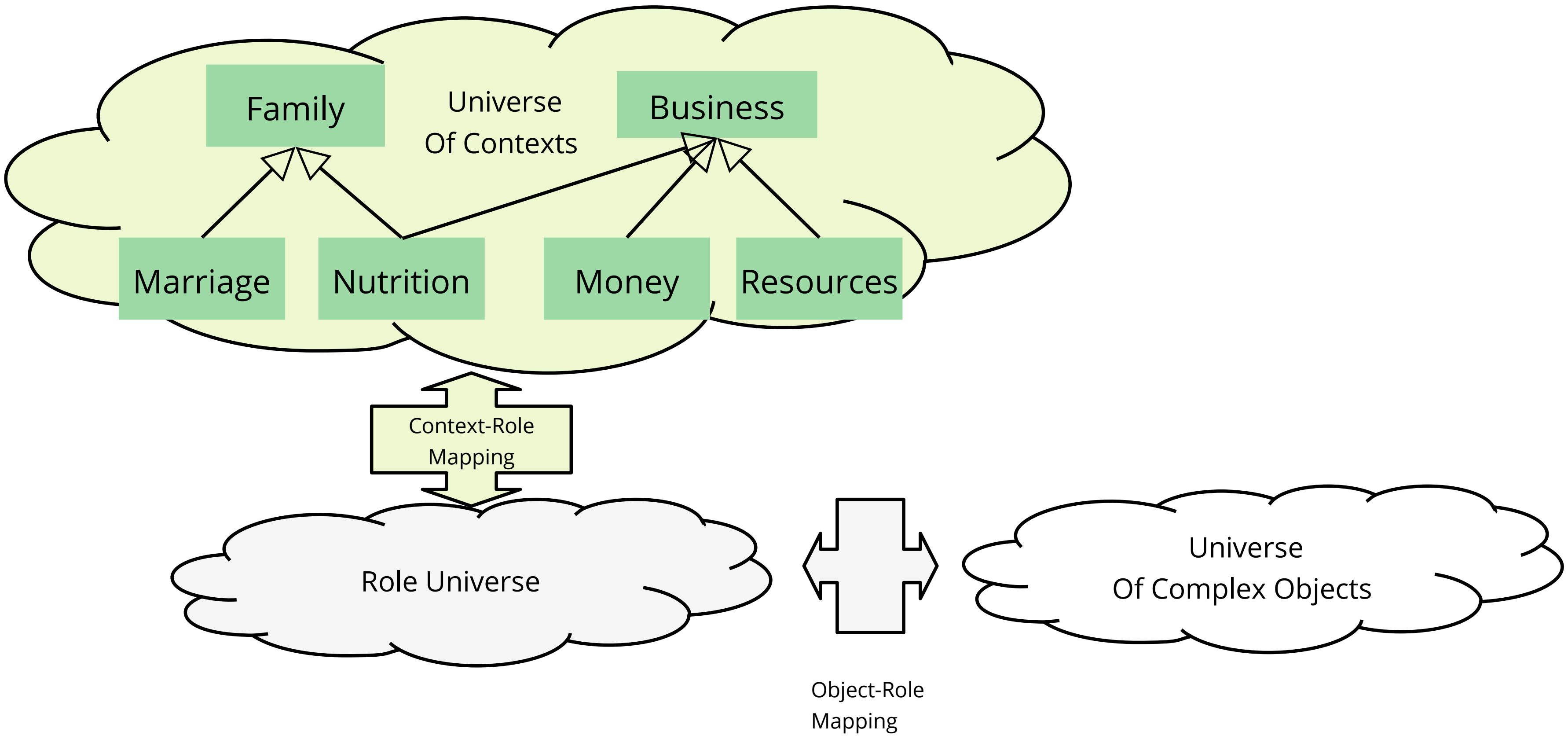
More on Concern-Separated Representation of Object Nets

- Compartments contain collaborations
- Compartments form ***indices*** to interprocedural slices

Photo by Bruno Kelzer on Unsplash



Example of Compartment Multi-Hierarchies



TECHNISCHE UNIVERSITÄT DRESDEN

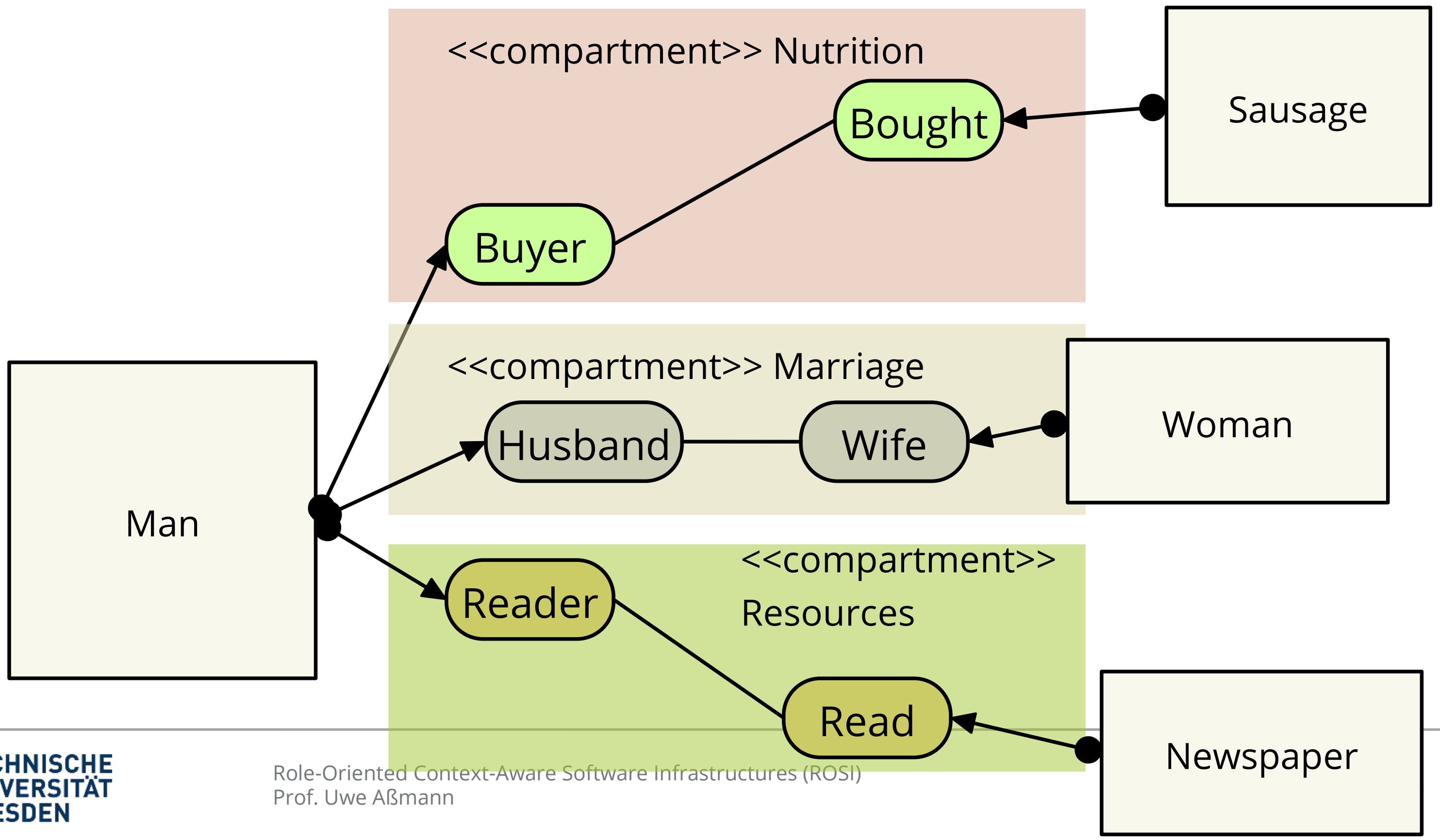
DRESDEN concept

Role-Oriented Context-Aware Software Infrastructures (ROSI)

1.3. Advantages of Roles: Simple Static and Dynamic Data Extensibility

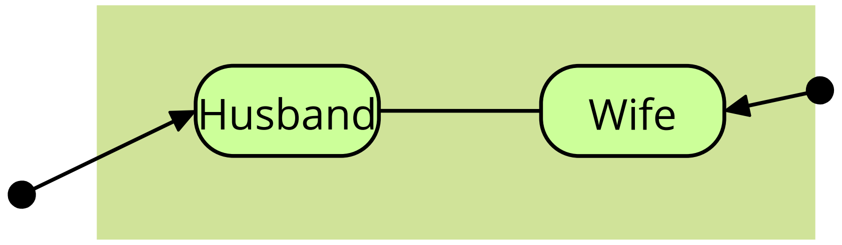
Simplified Extension with Compartments

- Object-role nets can be *extended by* new compartments with new role models collaborations

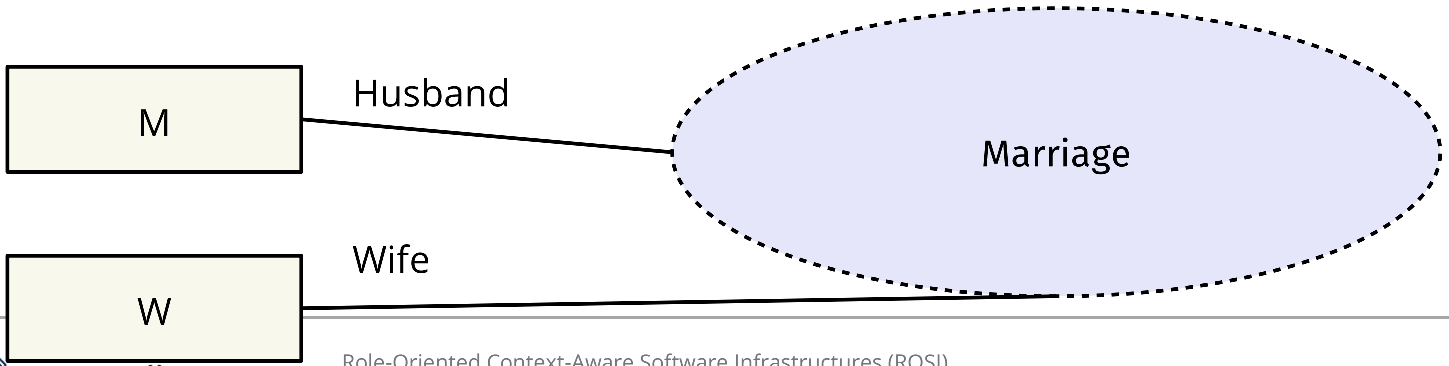


A Compartment is a Relational Module (Collaboration)

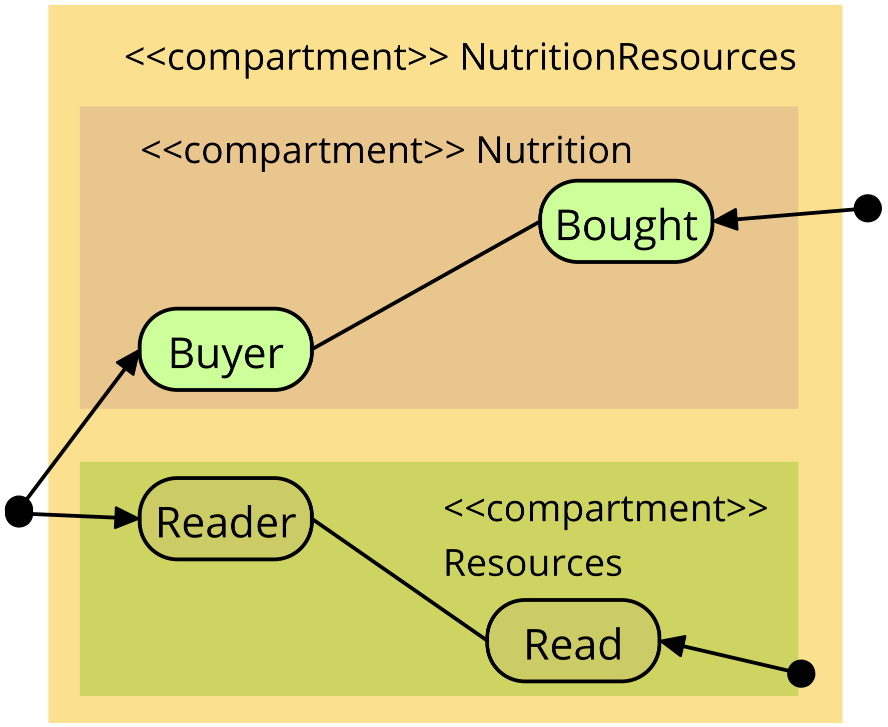
- Nets of roles with open ends, open *plays-a* tentacles,
  - to be attached to object cores



- UML Notation (class level) with *role-type parameter* P:



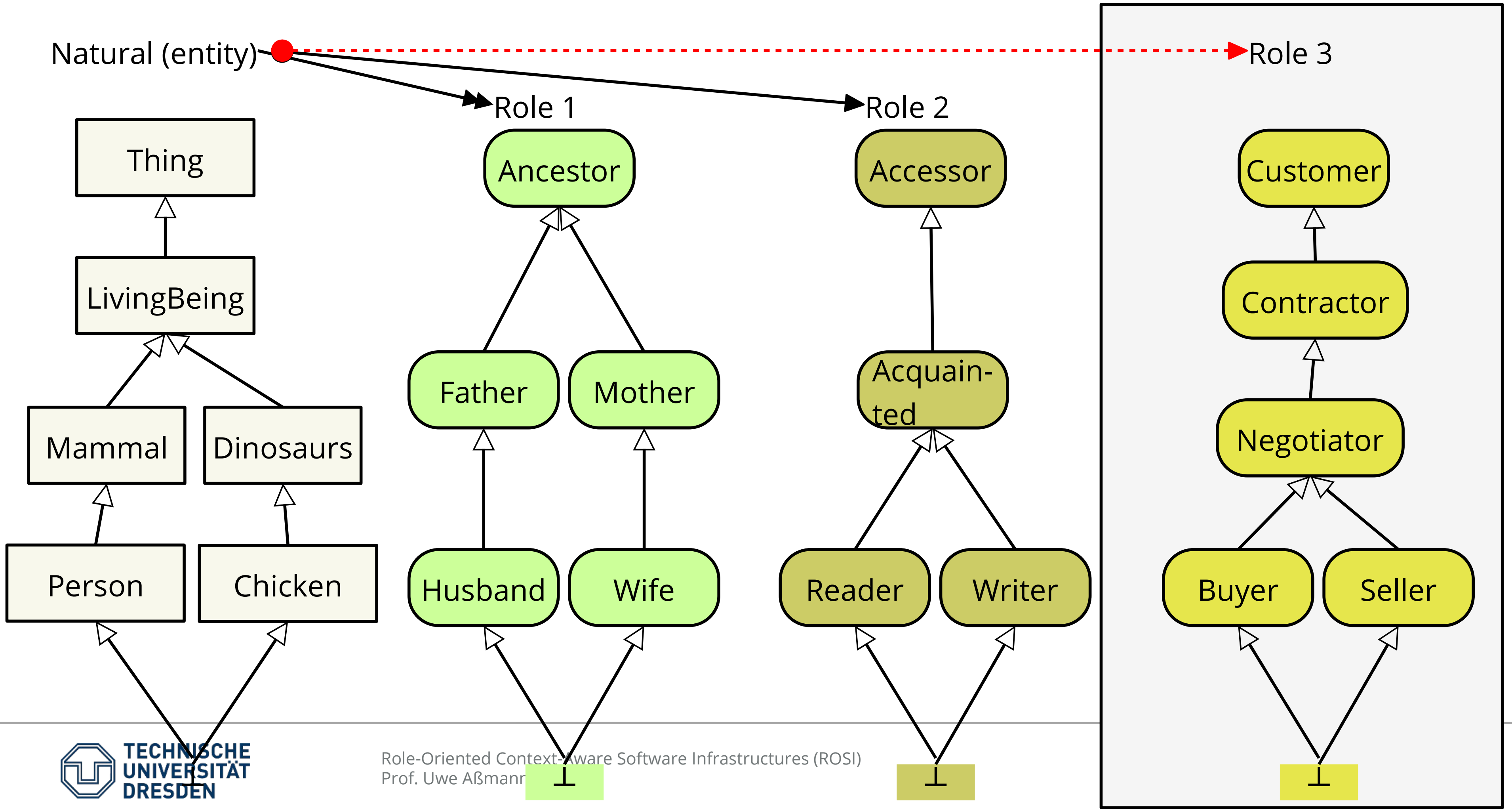
Structured Compartment: Resources and Nutrition



<<compartment>> NutritionResources
<<compartment>> Nutrition
Bought
Buyer
Reader
<<compartment>> Resources
Read

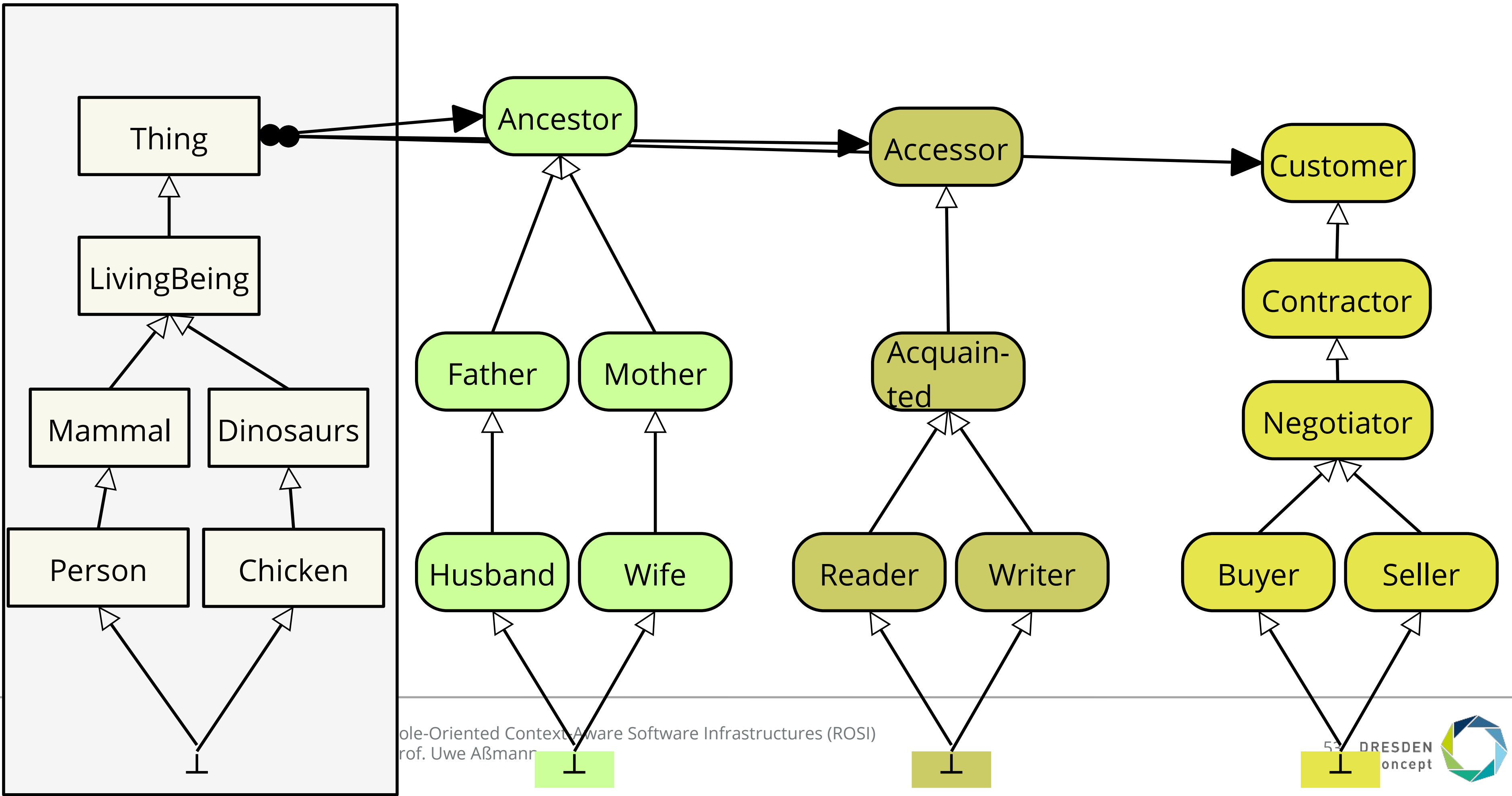
Extension on the Steimann Product Lattice

- A new role relationship extends the product lattice by another dimension.



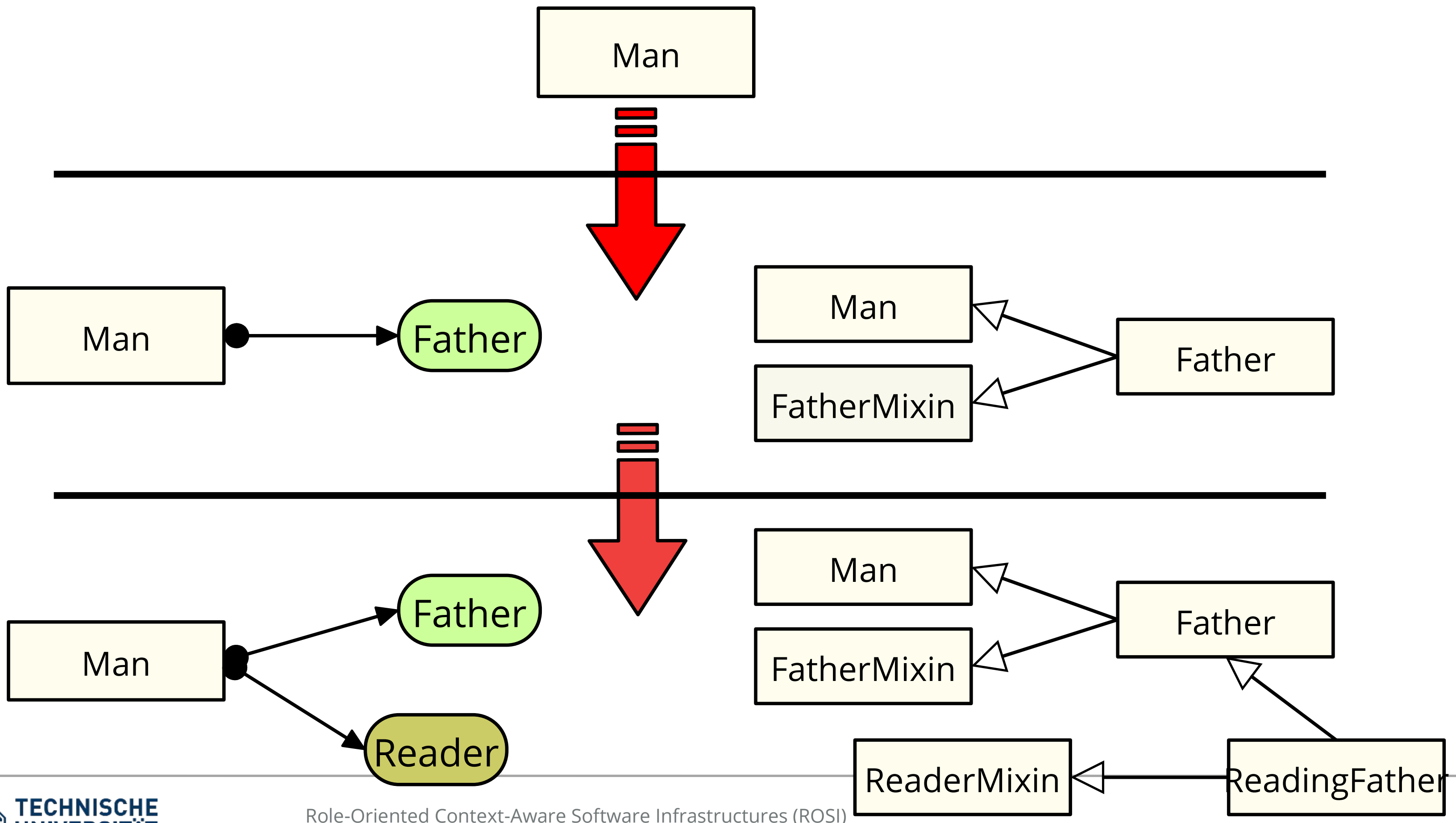
Separation of Concerns with Roles: Identity of Objects is Fixed to Core Facet of Product Lattice

- Role type extensions does not change the name of the core type nor of the full type (polymorphism)

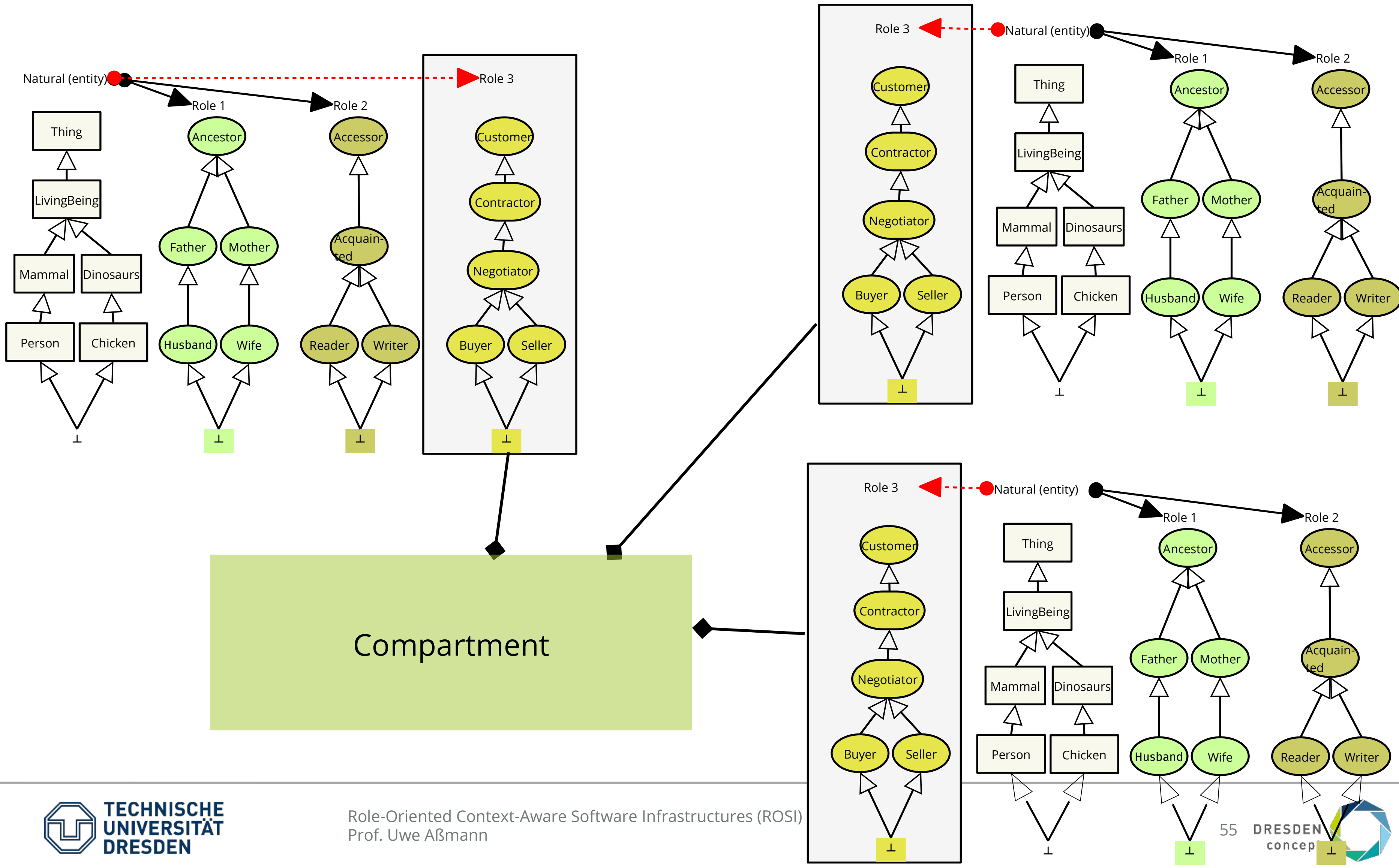


Separation of Concerns with Roles: Simplifies Inheritance Hierarchies

- Role Extension Retains Core Identity



Compartment Superimposition extends the Steimann Lattices of all involved Classes

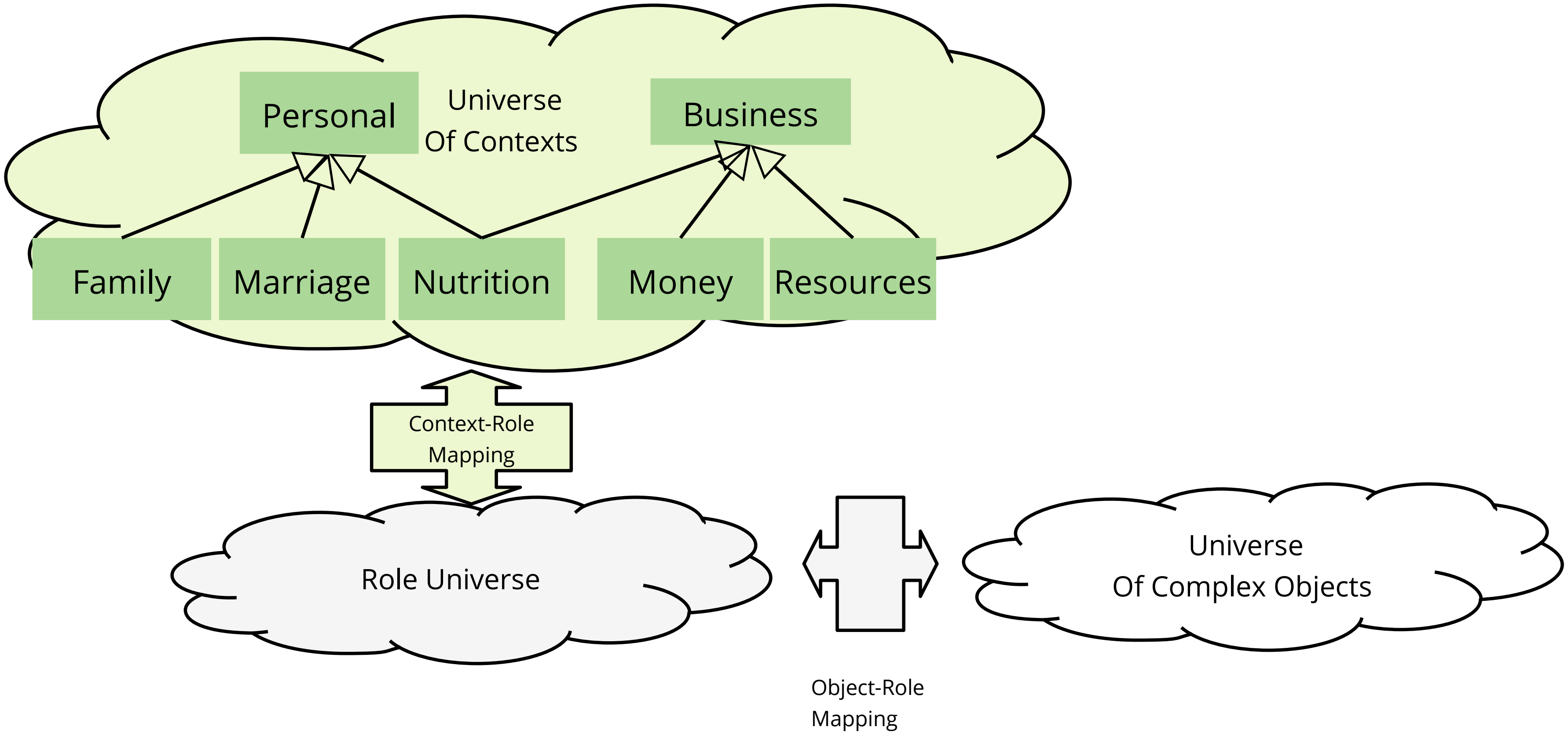


Extension and Adaptatation in the Steimann Lattice Retains Inheritance

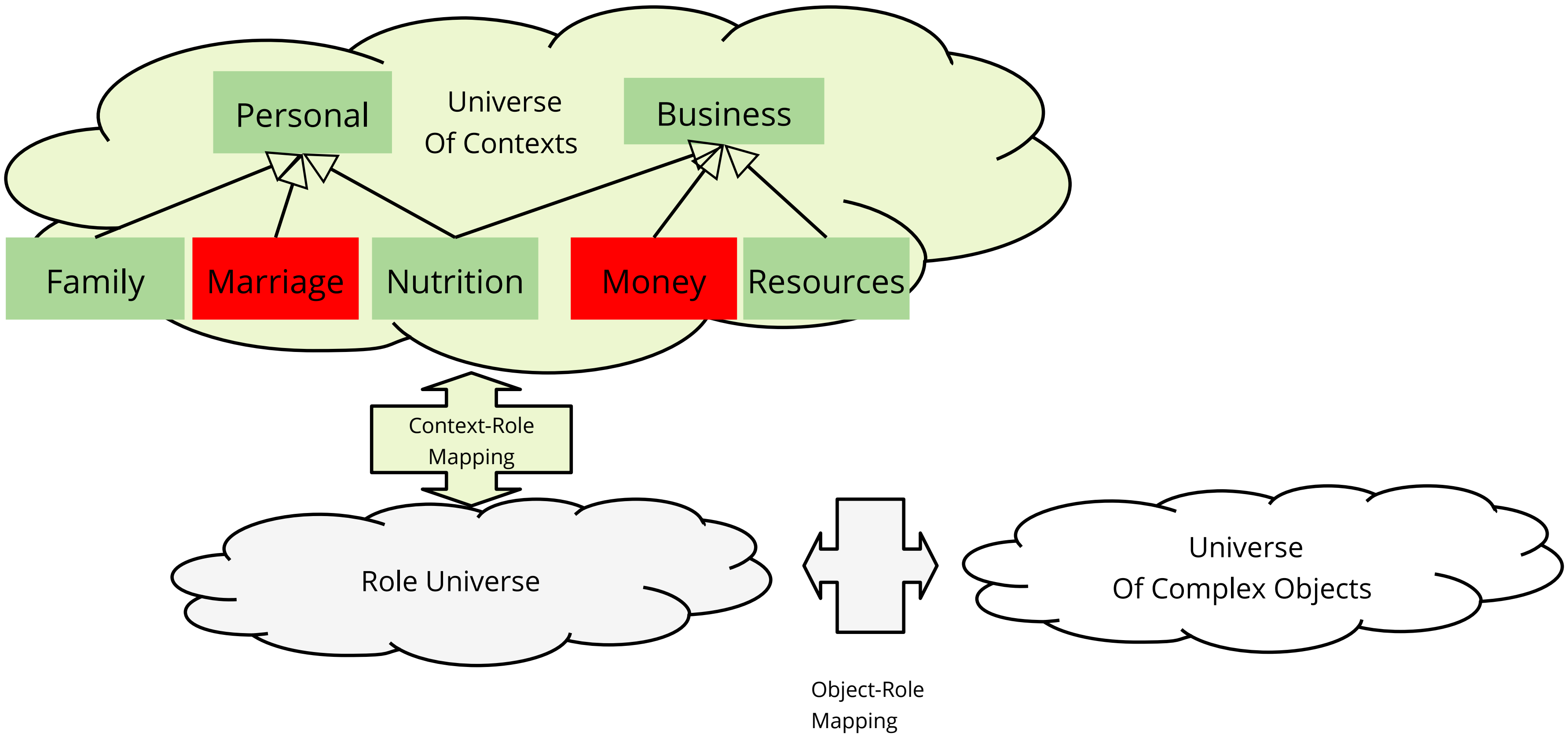
- Stable entity inheritance hierarchies, if concepts are added *relationally* to a model
  - Otherwise: extension of superclasses necessary (role classes become superclasses of entity classes)
  - Adding of new *concerns* is simple (adding a collaboration)

Superimposition of compartments to objects in Steimann-factored form retains all inheritance structures

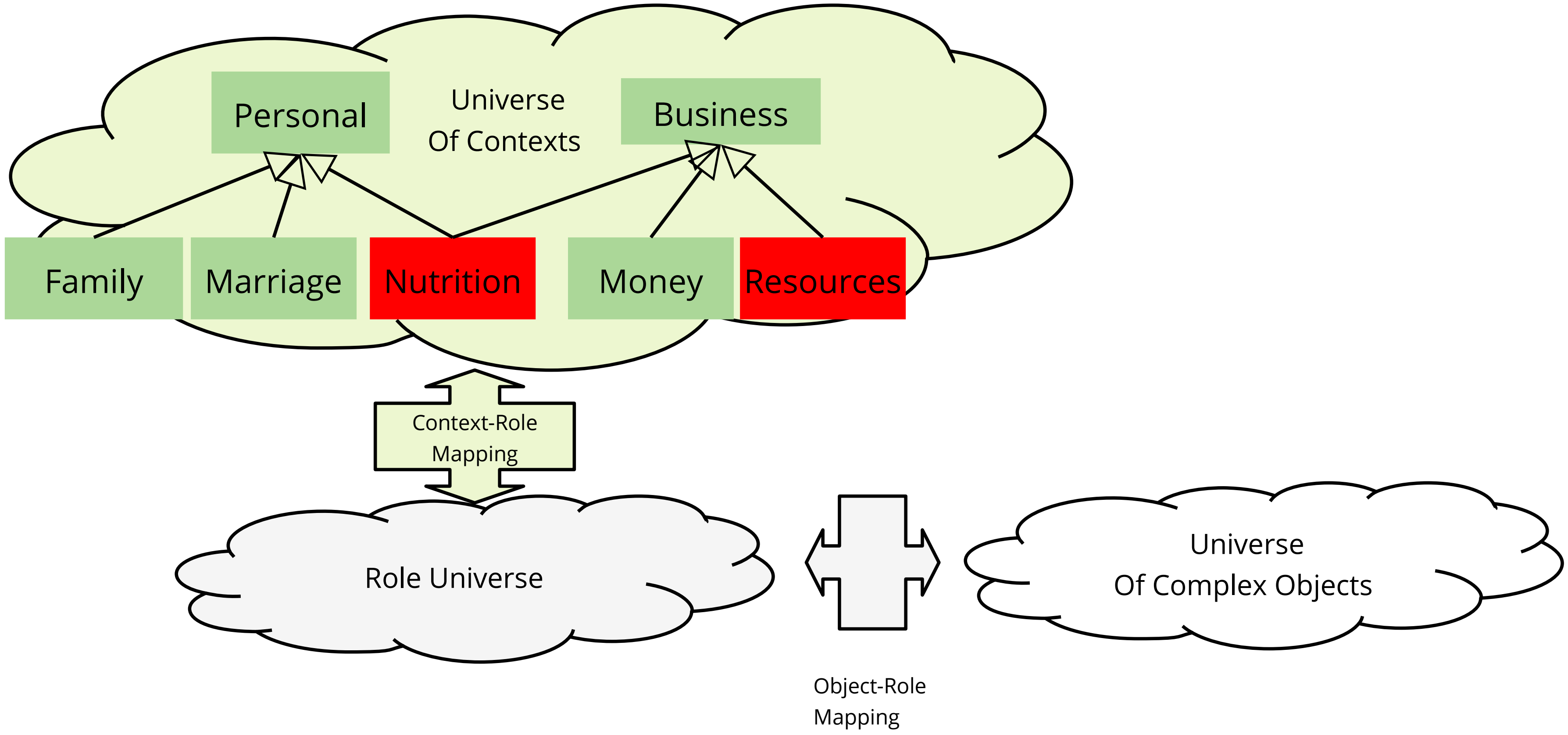
Adaptability with Compartment Multi-Hierarchies



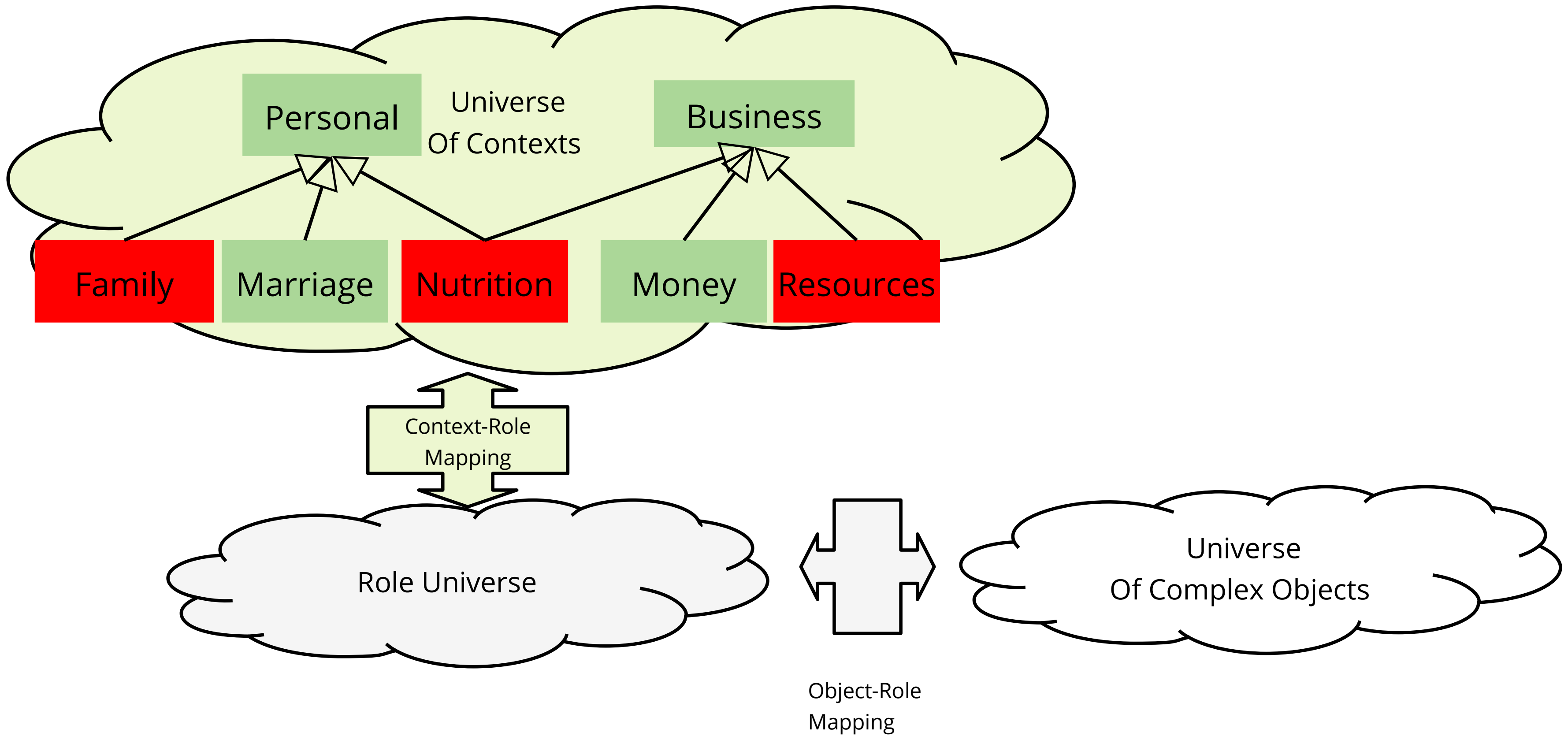
Adaptability with Compartment Multi-Hierarchies



Adaptability with Compartment Multi-Hierarchies



Adaptability with Compartment Multi-Hierarchies



ROSI Programming with SCROLL

- Compartment and Role Classes
- Dynamic Role Playing with *deep roles*
- SCROLL Scala Library https://github.com/max-leuthaeuser
- Change of context means to change to a new variant of the software
- SCROLL is perfect for *dynamic software product lines (DSPL)*

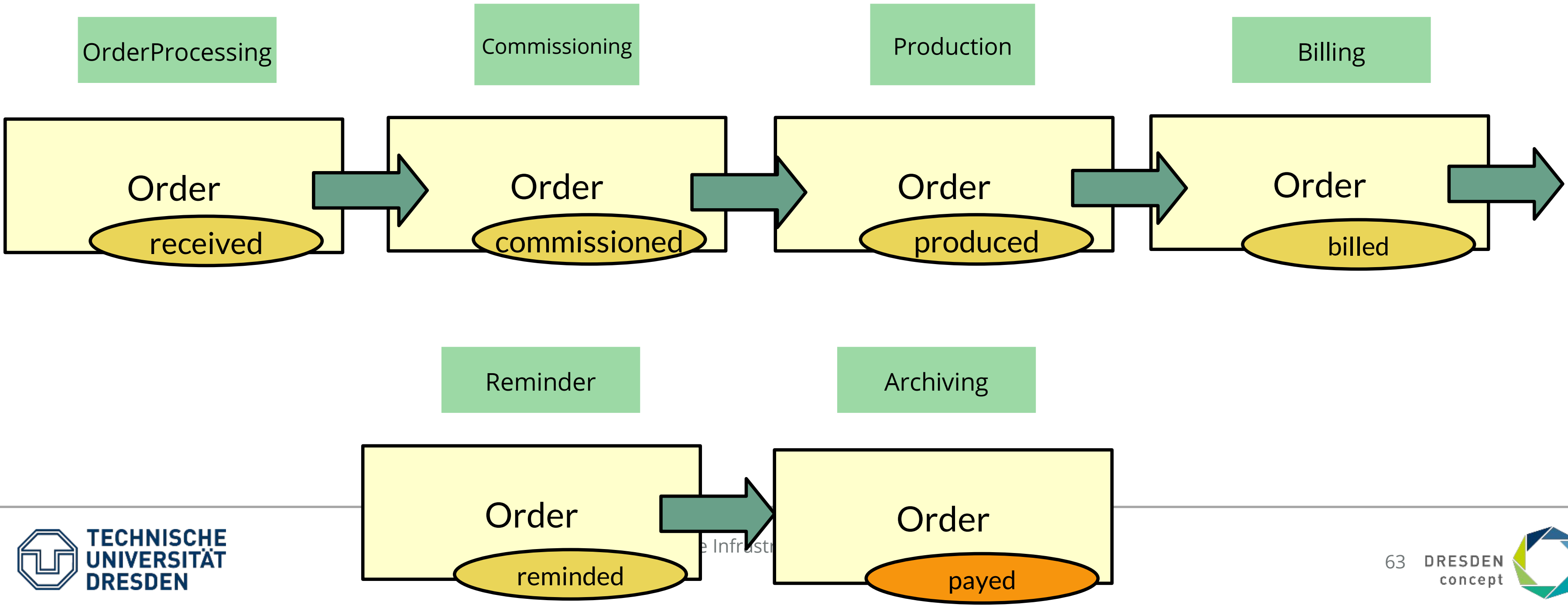
Roles and context are ready for programming in SCROLL

Role-Oriented Context-Aware Software Infrastructures (ROSI)

1.4. Roles and their Benefit for Separation of Concerns

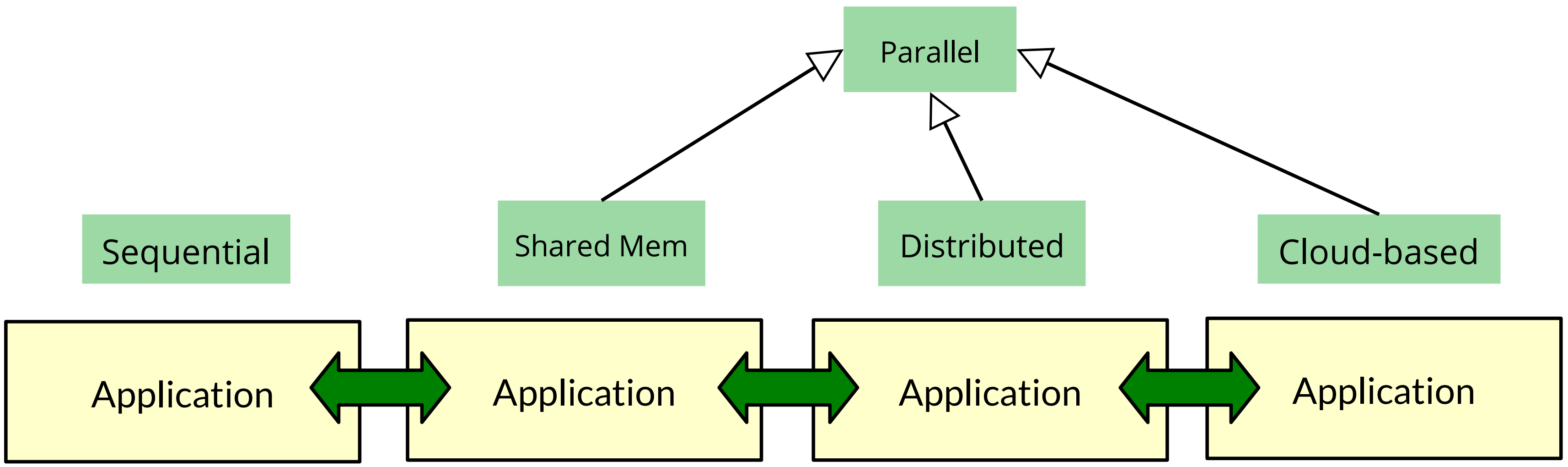
Business Objects with Roles and Contexts

- In large ERP frameworks (see SAP) business objects get very complex
- Ex.: **Order** gets different contexts, with roles
  - Every phase defines a context with different collaborators
- Dynamic Extensibility and Variability (Adaptation) by activation of new contexts



Parallel Objects with Roles and Contexts

- Selection of synchronisationprotocol by activation of new contexts
  -

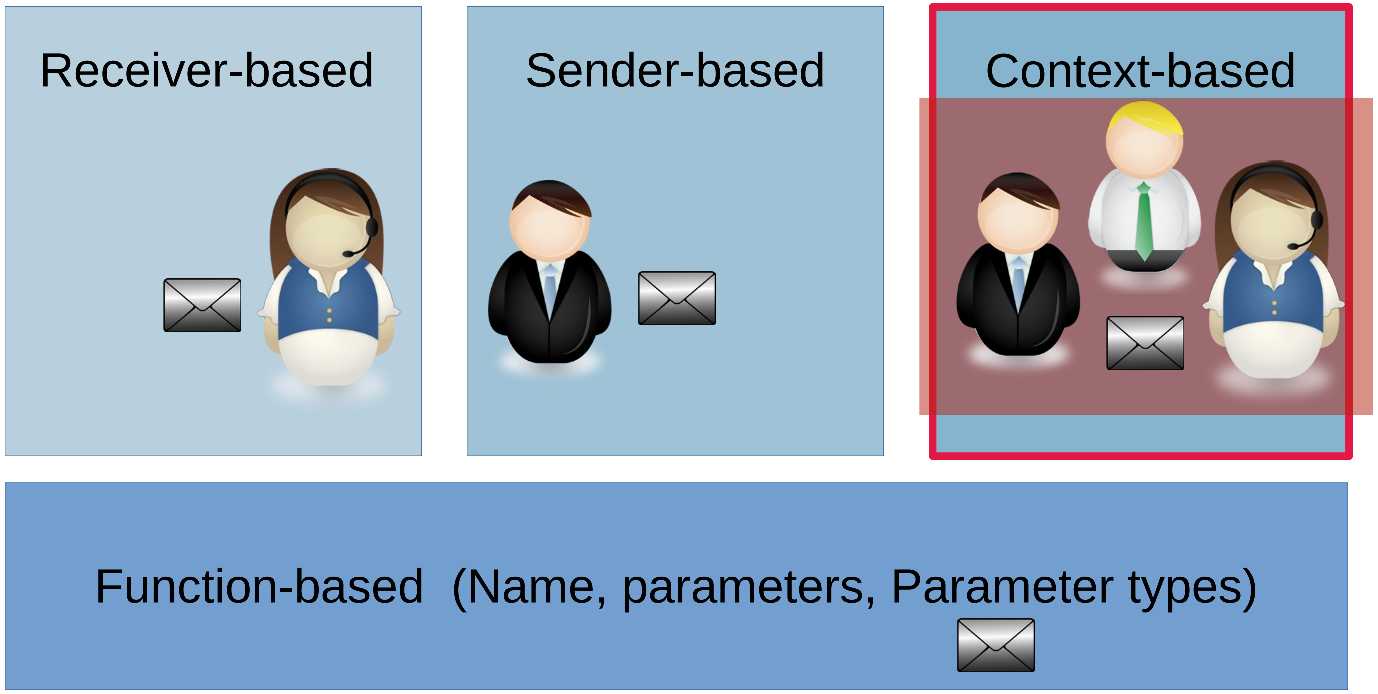


Advantages of ROSI for System Construction

- Separation of Concerns
  - Natural features – Context-dependent features
  - Dynamic features – static features
- Representation of roles as interprocedural graph slices
- Adaptability
  - Extensibility
  - Aspect Orientation (behavioral extensibility)
  - Variability (delayed role embedding decisions)
  - Substitutability (of roles and role models)

ROSI supports Roles and Contexts for Multi-Dimensional Dispatch for Multi-Polymorphism

- How is the semantics of a function (method) determined?



Dijkstra on Separation of Concerns

E. W. Dijkstra “On the Role of Scientific Thought”, EWD 447 Selected Writings on Computing: A Personal Perspective, pages 60–66, 1982.

"Let me try to explain to you, what to my taste is ***characteristic for all intelligent thinking.***

It is, that one is willing to study in depth **an aspect of one's subject matter in isolation** for the sake of its own consistency, all the time knowing that one is occupying oneself only with one of the aspects.

We know that a program must be correct and we can study it from that viewpoint only; we also know that it should be efficient and we can study its efficiency on another day, so to speak. In another mood we may ask ourselves whether, and if so: why, the program is desirable. But nothing is gained --on the contrary!-- by tackling these various aspects simultaneously.

Roles and contexts introduce separations of concerns.

Intelligent thinking and scientific thought

It is what I sometimes have called **"the separation of concerns"**, which, even if not perfectly possible, is yet the only available technique for effective ordering of one's thoughts, that I know of.

This is what I mean by **"focussing one's attention upon some aspect"**: it does not mean ignoring the other aspects, it is just doing justice to the fact that from this aspect's point of view, the other is irrelevant. It is being one- and multiple-track minded simultaneously.

Scientific thought comprises "intelligent thinking" as described above. A scientific discipline emerges with the --usually rather slow!-- discovery of which aspects can be meaningfully **"studied in isolation for the sake of their own consistency"**, in other words: with the discovery of useful and helpful concepts. Scientific thought comprises in addition the conscious search for the useful and helpful concepts.

The End

https://rosi-project.org

Important References

- T. Reenskaug, P. Wold, and O. Lehne. Working with Objects, The OOram Software Engineering Method. Manning Publications, 1996.
- Friedrich Steimann. On the representation of roles in object-oriented and conceptual modelling. Data Knowl. Eng, 35(1):83-106, 2000.
- Friedrich Steimann. A radical revision of UML's role concept". UML 2000, 3rd International Conference, Springer LNCS, 194–209.
  - and many more, see his home page at U Hagen
- Charles W. Bachman and Manilal Daya. The role concept in data models. In VLDB '1977: Proceedings of the third int.l conf. on Very large data bases, pages 464–476. VLDB Endowment, 1977.

Ontological Foundations of Metatypes

- Giancarlo Guizzardi, Heinrich Herre, and Gerd Wagner. On the general ontological foundations of conceptual modeling. 21st Int. Conf. on Conceptual Modeling (ER 2002), LNCS 2503, pages 65-78, 2002.
- Guizzardi, G. (2005). Ontological Foundations for Structural Conceptual Models. PhD thesis, University of Twente.
- [Guariono] Nicola Guarino Chris Welty. Supporting ontological analysis of taxonomic relationships. Data and Knowledge Engineering, 39:51--74, 2001.
- Paul Lorenzen, Oswald Schwemmer. Konstruktive Logik, Ethik und Wissenschaftstheorie. BI Hochschultaschenbücher, Band 700, 1973.
- Paul Lorenzen. Lehrbuch der konstruktiven Wissenschaftstheorie, Metzler Reprint, 2000.
- H. Wedekind, E. Ortner, R. Inhetveen. Informatik als Grundbildung. Informatik Spektrum, Springer, April 2004
- H. v. Braun, W. Hesse, H.B. Kittlaus,G. Scheschonk. Ist die Welt objektorientiert? Von der natürlich-sprachlichen Weltsicht zum OO-Modell. Uni Marburg.

Programming Languages

- S. Herrmann. Object teams: Improving modularity for crosscutting collaborations. In Proc. Net Object Days 2002, 2002.
- S. Herrmann. A precise model for contextual roles: The programming language objectteams/java. Applied Onthology, (to appear), 2007.
- www.objectteams.org: a programming lanugage with roles
- Max Leuthäuser. A Pure Embedding of Roles - Exploring 4-dimensional Dispatch for Roles in Structured Contexts. PhD thesis, Technische Universität Dresden, August 2017.
  - This PhD thesis developes a programming language for contexts and roles, based on some implementation patterns and the base language Scala. " http://nbn-resolving.de/urn:nbn:de:bsz:14-qucosa-227624
- [SCROLL] SCROLL Library https://github.com/max-leuthaeuser

Important References

- D. Bäumer, D. Riehle, W. Silberski, and M. Wulf. Role object. In Conf. On Pattern Languages of Programming (PLOP), 1997.
- Dirk Riehle and Thomas Gross. Role model based framework design and integration. ACM SIGPLAN Notices, 33(10):117-133, October 1998.
- Dirk Riehle. Framework Design - A Role Modelling Approach. PhD thesis, ETH Zürich, 2000. No. 13509. www.riehle.org.
- Y. Smaragdakis and D. Batory. Mixin layers: an object-oriented implementation technique for refinements and collaboration-based designs. ACM Transactions on Software Engineering and Methodology, 11(2):215–255, 2002.

Works at TU Dresden

- Thomas Kühn. A Family of Role-Based Languages. PhD thesis, Technische Universität Dresden, March 2017. http://nbn-resolving.de/urn:nbn:de:bsz:14-qucosa-228027
- U. Aßmann, S. Zschaler, and G. Wagner. Ontologies, Meta-Models, and the Model-Driven Paradigm, Handbook on Ontologies and Software Engineering. pages 249–273. Springer, 2006.
- U Aßmann, J Johannes, J Henriksson, and Ilie Savga. Composition of rule sets and ontologies. In F. Bry, editor, Reasoning Web, Second Int. Summer School 2006, number 4126 in LNCS, pages 68-92, Sept 2006. Springer.
- M. Pradel, J. Henriksson, and U. Aßmann. A good role model for ontologies: Collaborations. Int. Workshop on Semantic-Based Software Development. at OOPSLA'07, Montreal, Oct 22, 2007.
- Christian Piechnick, Sebastian Richly, Sebastian Götz, Claas Wilke, and Uwe Aßmann. Using Role-Based Composition to Support Unanticipated, Dynamic Adaptation - Smart Application Grids. In Proceedings of ADAPTIVE 2012, The Fourth International Conference on Adaptive and Self-adaptive Systems and Applications, pages 93-102, 2012.

Works at TU Dresden

- J. Reimann, M. Seifert, U. Aßmann. Role-based generic model refactoring. MODELS Okt. 2010
- Thomas Kühn, Max Leuthäuser, Sebastian Götz, Christoph Seidl, and Uwe Aßmann. A metamodel family for role-based modeling and programming languages. In Benoit Combemale, David J. Pearce, Olivier Barais, and Jurgen J. Vinju, editors, SLE, volume 8706 of Lecture Notes in Computer Science, pages 141--160. Springer, 2014.
- Thomas Kühn, Stephan Böhme, Sebastian Götz, and Uwe Aßmann. A combined formal model for relational context-dependent roles. In Richard F. Paige, Davide Di Ruscio, and Markus Völter, editors, SLE, pages 113--124. ACM, 2015.
- Johannes Mey, René Schöne, Görel Hedin, Emma Söderberg, Thomas Kühn, Niklas Fors, Jesper Öqvist, and Uwe Aßmann. Continuous model validation using reference attribute grammars. In Proceedings of the 11th ACM SIGPLAN International Conference on Software Language Engineering, SLE 2018, pages 70--82, New York, NY, USA, 2018. ACM.

Other PhD Theses (all available via www.qucosa.de)

- Mirko Seifert. Designing Round-Trip Systems by Model Partitioning and Change Propagation. PhD thesis, Dresden University of Technology, June 2011.
  - Shows how roles simplify round-trip engineering by partitioning data
- Sebastian Richly. Autonom rekonfigurierbare Workflows. PhD thesis, Dresden University of Technology, December 2011.
  - shows how roles can be used to provide an extensible tool platform
- Christian Wende. Language Family Engineering. PhD thesis, Dresden University of Technology, March 2012.
  - shows how roles can be used to do context-based language composition