


Reutlingen
University

30.05. – 03.06.2021, DBKDA 2021


Information Integration using the Typed Graph Model



**Fritz Laux, Prof. emeritus,
Reutlingen University,
Reutlingen, Germany,
fritz.laux@reutlingen-university.de**

**Malcolm Crowe, Prof. emeritus,
University of the West of Scotland,
Paisley, UK,
malcolm.crowe@uws.ac.uk**

© F. Laux




We call information integration the integration of data and their related schemata.
This has been a challenge for more than 40 years.

But, in the meantime we have a better understanding due to more theoretical
research and practical work in this area.

My talk is about “How do we get from different data sources to an integrated
schema”.

My name is Fritz Laux. I’m a retired professor from Reutlingen University where I
was responsible for database teaching and research since the start of our
department.

This work was prepared in collaboration with Professor Malcolm Crowe from the
UWS.

 Reutlingen University Aim Motivation Example TGM Workflow Patterns Quality Lessons References 2 / 16 © F. Laux	Aim of the Talk
	<p>⇒ <i>Propose a Framework for information integration and motivate the use of the Typed Graph Model (TGM)</i></p> <p>⇒ Contents</p> <ul style="list-style-type: none"> ⇒ Present the TGM with relevant properties for our purpose ⇒ Introduce an integration framework <ul style="list-style-type: none"> ⇒ Using the TGM as intermediate supermodel ⇒ Introduce some mapping patterns ⇒ Present practical guidelines and quality criteria for the mapping <p>⇒ Research challenges</p> <ul style="list-style-type: none"> ⇒ Generate intermediate Typed Graph Schemata (TGS) ⇒ (Semi-)automate the matching ⇒ Ensure high-quality mapping

What can you expect in the next 15 minutes?


I will give a short **motivation** for **information integration** and present the use of the **Typed Graph Model** (TGM for short) for visualizing the schemata and its mapping.

Next, the integration framework is introduced and some mapping patterns together with guidelines are presented.

The **talk is built around a scenario** that serves as example for the information integration process.

There still exist some challenges like the automated intermediate schema generation and matching, and not all are solved yet.

At least I can present some guidelines for high-quality integration and a theorem for a **“perfect” mapping**.



Reutlingen
University

Aim

Motivation

Example

TGM

Workflow

Patterns

Quality

Lessons

References

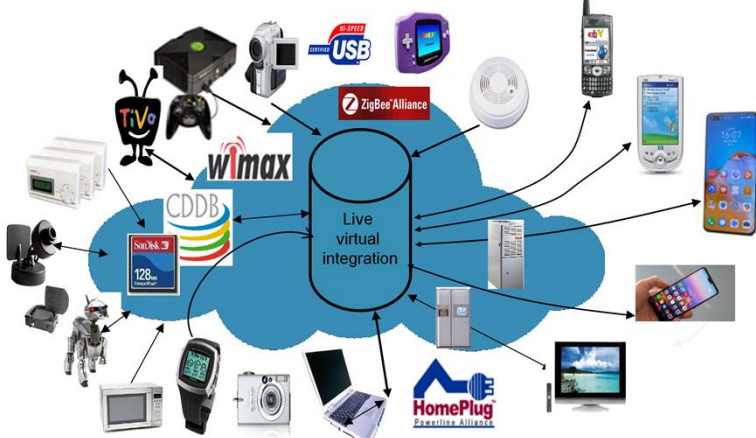
3 / 16

© F. Laux

Motivation for Data Integration

➤ *There is a need to integrate heterogeneous data sources to...*

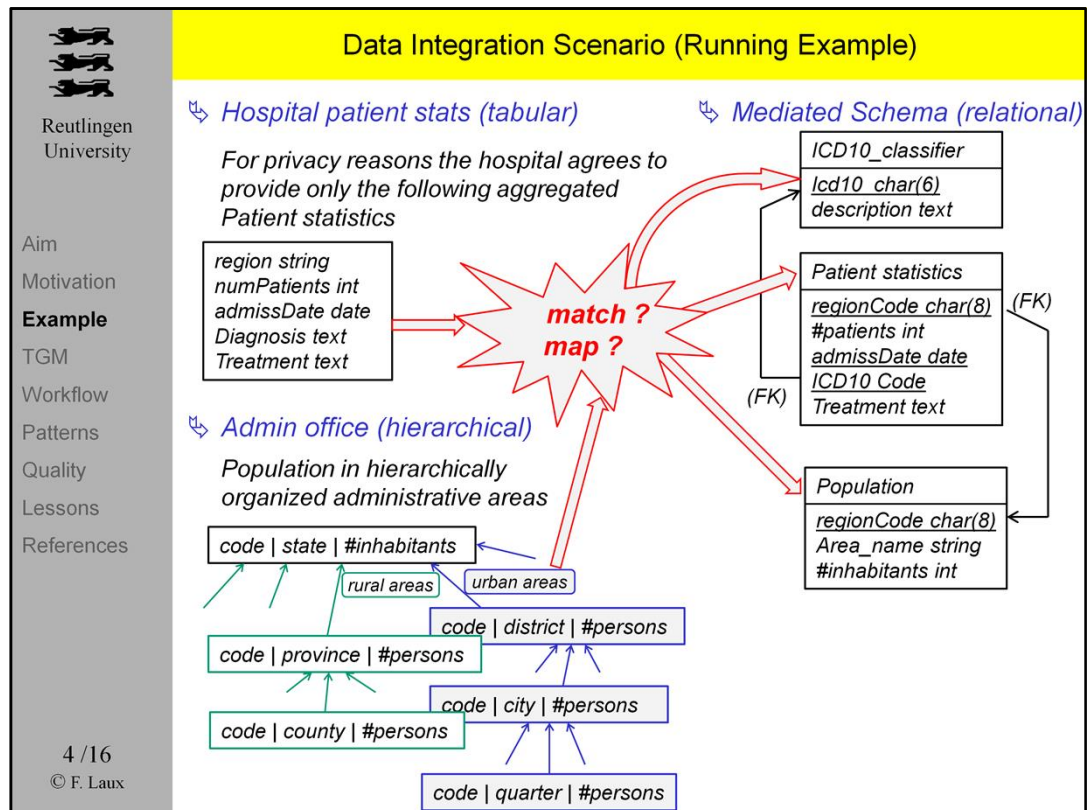
- ☞ gain added value (knowledge, insights) through virtual data integration for decision support, predictive analysis, performance management, etc.
- ☞ coordinate complex processes in (near) real-time (e.g. traffic control, industry 4.0, **fight epidemic**)



The need to integrate different data sources is rather obvious.

We expect to gain new knowledge and insights to our data that help us to make predictive analysis, coordinate complex processes and so on.

Many of the control decisions need up-to-date high quality information from different sources, not only in industry production but also in emergency situations like fighting epidemic, earthquake or other natural disasters.



Let's illustrate our approach with an example scenario that we are going to solve during this talk:

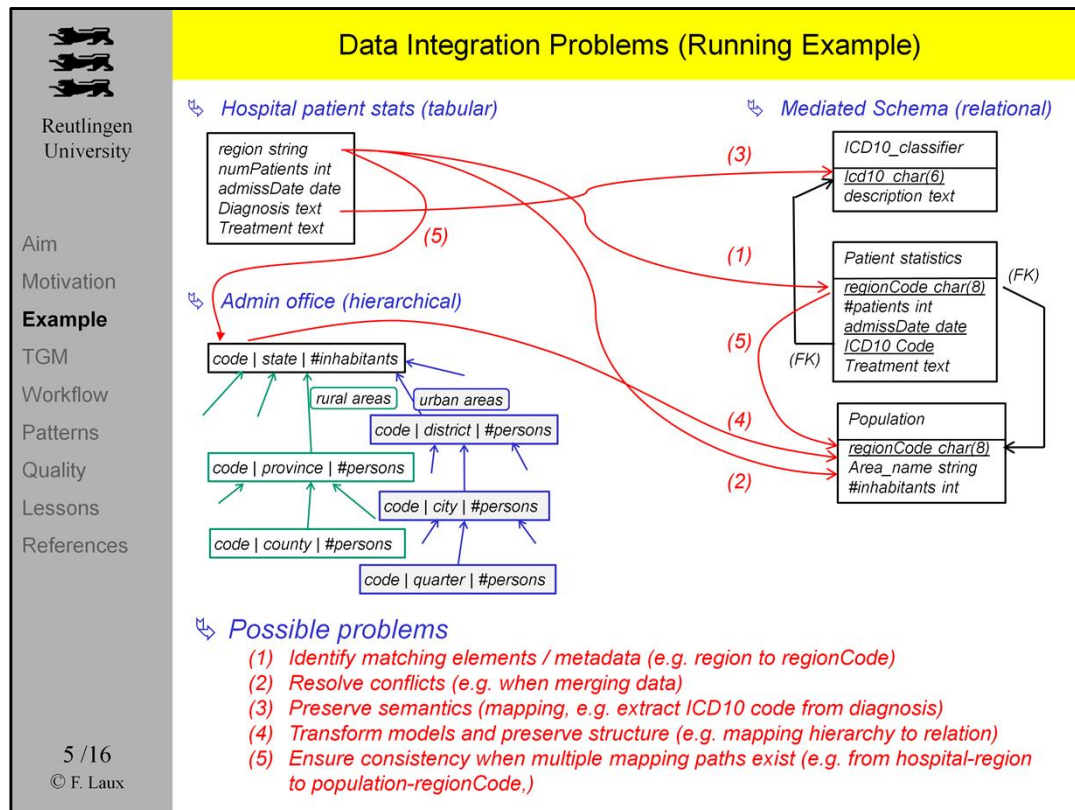
The **World Health Organization** is monitoring world health situation and reports on emergencies and epidemics. The data is provided in many different formats by national or regional autonomous actors like national administrations or hospitals.

For simplicity, we only show one **hospital** providing aggregated tabular data of patients grouped by regions, admission date and diagnosis.

The **statistics admin offices** delivers demographic data in an hierarchical structure reflecting the administrative areas.

We want to create a **mediated schema** that combines information from both sources using the **international classifier for diseases (ICD)**.

Given an Integration Schema, the question is, how do we find matches for the data items and how to transform them.




We can identify 5 possible problems in this example: How do we ...

- (1) Map regions to code
- (2) Resolve conflicts when merging data (for instance: region to area code)
- (3) Extract ICD data from the diagnosis
- (4) Transform models and preserve structure of the geographic Code
- (5) ensure that the Population data are consistent

So far we used different graphical representations for the schemata.

But, to solve the matching and mapping problem a **uniform and consistent** graphical **representation** would be **helpful**.

We have chosen the Typed Graph Model (TGM) to act as **intermediate supermodel** that can help us to deal with the above mentioned questions when integrating the schemata.



Reutlingen
University

Aim
Motivation
Example
TGM
Workflow
Patterns
Quality
Lessons
References

6 / 16
© F. Laux

Definition (simplified): The Typed Graph Model (TGM)

↪ A *typed graph data model* is a tuple $TGM = (N, E, TGS, \phi)$ where:

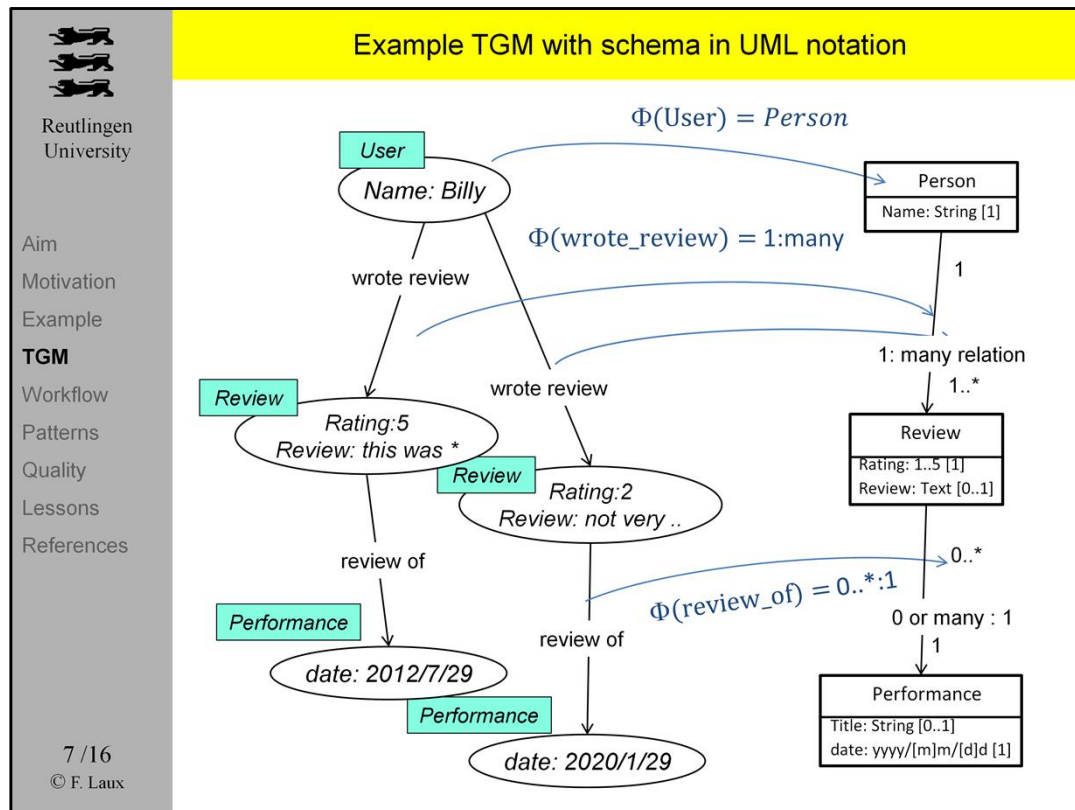
- ☞ N is the set of named (labeled) hyper-nodes n with data types from schema TGS.
- ☞ E is the set of named (labeled) hyper-edges e with properties of types from schema TGS.
- ☞ ϕ is a homomorphism that maps each node n and edge e to the corresponding type element of TGS.
- ☞ In addition TGS offers min-max multiplicity for each end of an edge plus integrity constraints C

↪ Using the UML notation for visualizing TGM (and TGS)

TGM	UML
$n \in N$	class
$e \in E$	association
min-max	(min .. max)
C	Constraints [...] or { ... }

- The **Typed Graph Model (TGM)** informally constitutes a property hyper-graph that conforms to a schema.
- It consist of typed hyper-nodes N and typed hyper-edges E both from a **Typed Graph Schema (TGS)**
- The homomorphism ϕ that maps each *instance* element to a *schema* element is essential for the integrity of the model.
- The Typed Graph Schema TGS offers min-max cardinality for each edge endpoint and supports additional integrity constraints.
- We use the UML class notation for visualizing nodes and UML associations for edges as it provides a compact rendering and extensions using constraints.

Next, we illustrate the use of TGM with a little example and show how it will enhance data quality by using types.

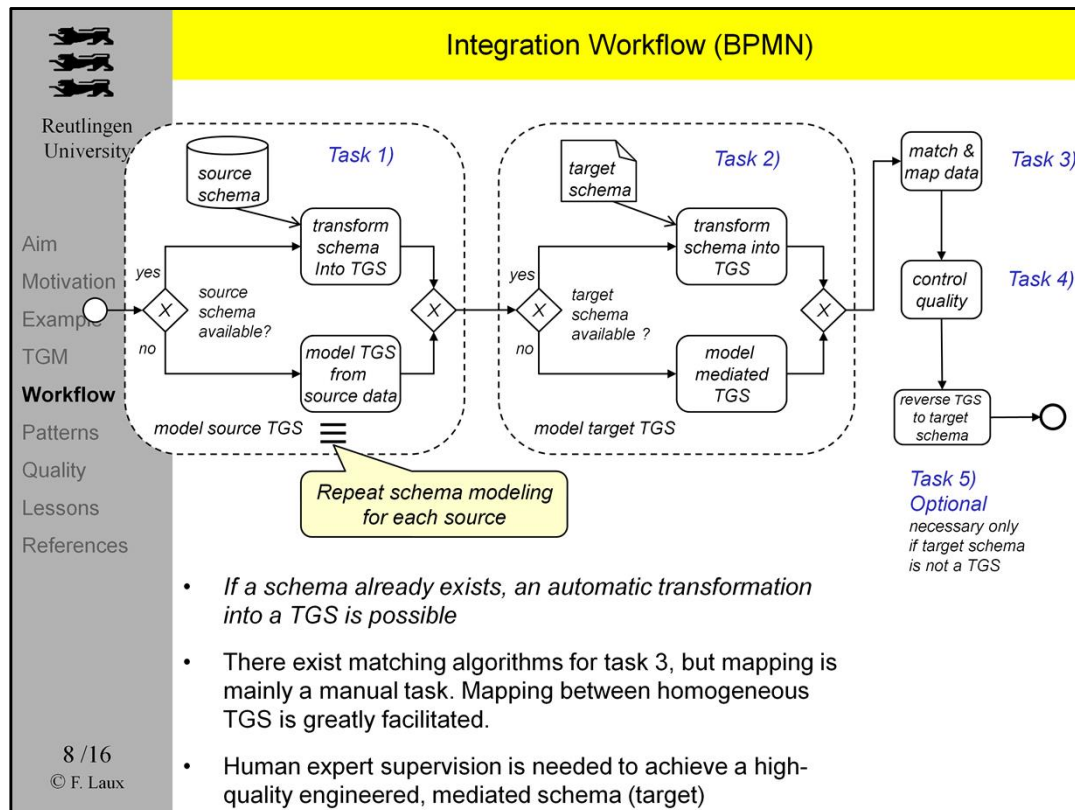


- On the left part of the picture we have an instance graph. It shows that Billy wrote 2 **Reviews**, one for each **performance**.
- The schema on the right uses UML for better visual clarity. It defines the allowed structure and allowed properties and labels of an instance graph.
- The function Φ maps the **User** to the data type **Person** which ensures that the **User** must have exactly one name. This is indicated by the number 1 in brackets. The **Review** itself is tied to the complex type **Review** with a mandatory Rating and an optional Review text. The **Performance** can have 2 properties, an optional title and a mandatory date. Even the date format is clearly prescribed by a format template.

The association cardinalities between **Person** and **Review** coerce that a Person has at least one Review. The mapping Φ ensures that there are no Users without Review.

The homomorphism Φ preserves structure between both graph levels. This means, that **wrote review** instances are tied to the 1 to many relation and therefore no second author is allowed to link to Billy's reviews.

A **Review** always refers to exactly one **Performance**, but, a **Performance** may have any number of **Reviews**, including none.



We are now ready to introduce the **Integration Workflow** .

The data integration workflow is shown as Business Process Model and Notation diagram and consists of five tasks:

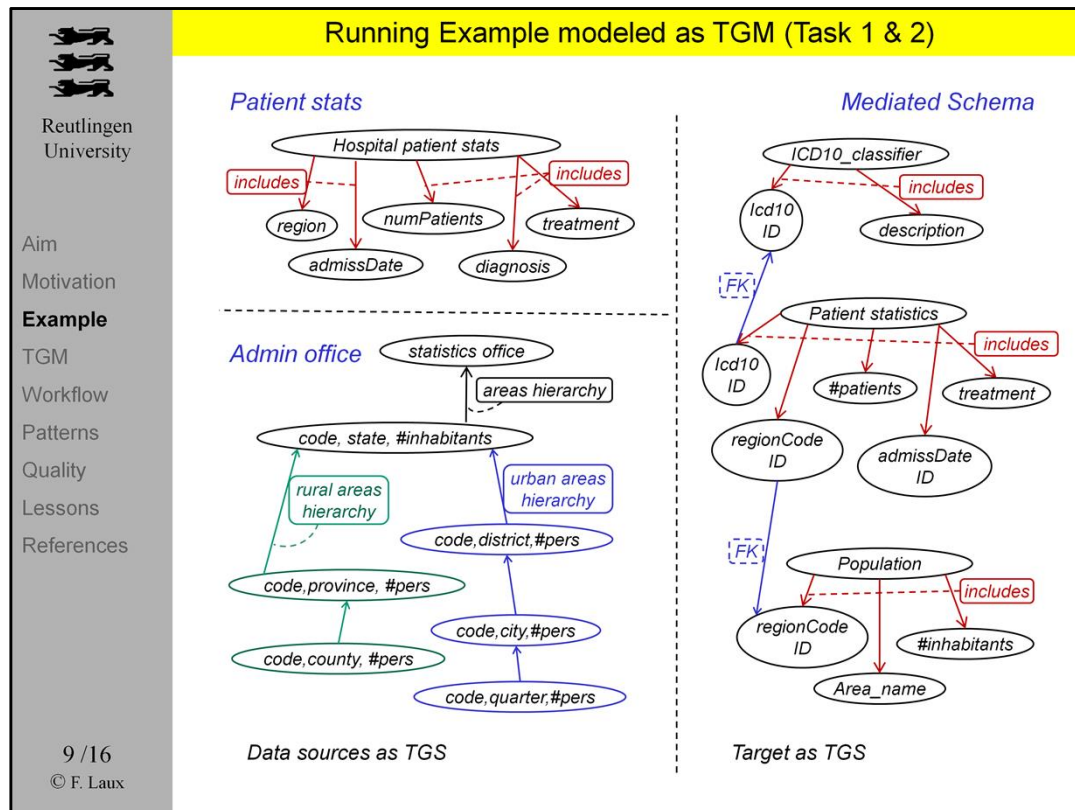
- 1) model all sources as Typed Graph Schema (TGS)
- 2) model the target as TGS
- 3) match and map the sources with the target TGS
- 4) check and improve the integration quality
- 5) convert the TGS back to the target again

The first and second tasks consist of two alternatives depending on the pre-existence of a schema. If a schema already exists an automatic transformation is possible. If a source has no schema, it is then necessary to collect structure and type information to model a TGS. This extra effort has the advantage to achieve a better data quality.

The 4th task of the workflow is crucial for *Enterprise Information Integration* and other data integration projects, which demand highly accurate information quality.

It might turn out that the resulting quality is not sufficient. As consequence, the process might have to be iterated with different mappings in order to improve the quality.

The last task is only necessary if the target schema is not a graph schema.



If we apply the workflow tasks 1 and 2 described on the previous slide to our running example, we get the following schemata.

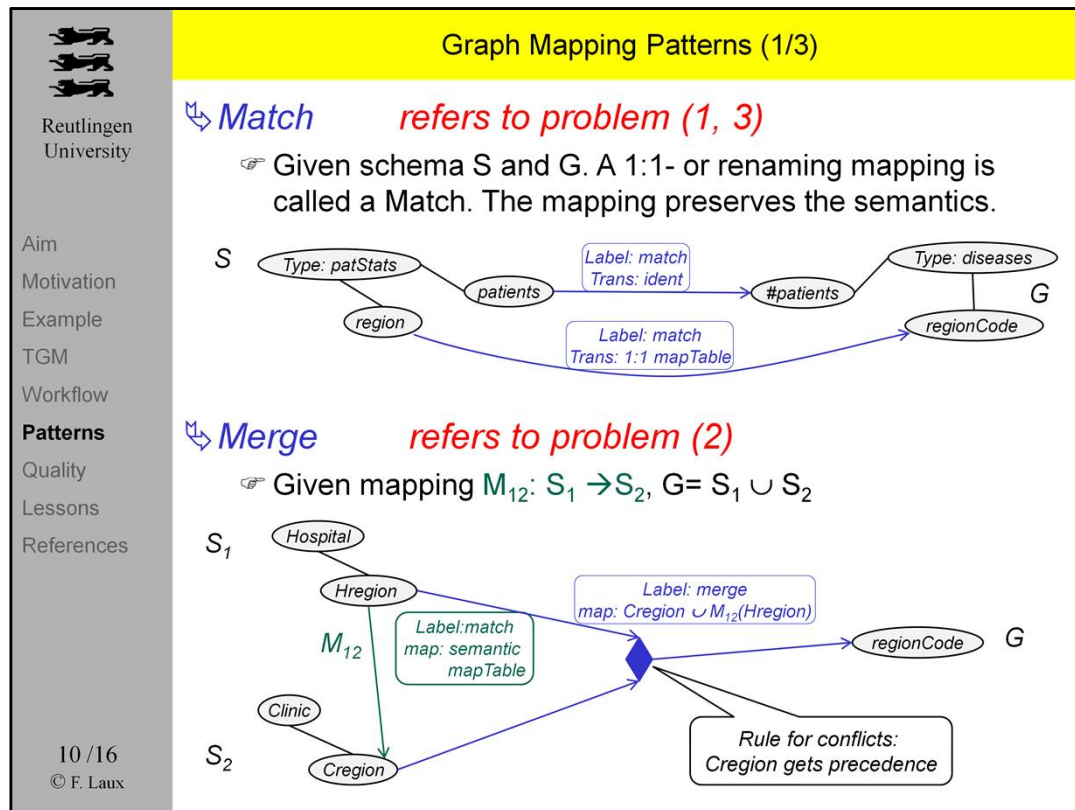
To not overload the picture the nodes are only visualized with property names inside the ovals. The data types are suppressed, but the edge types are color coded and connected by dashed lines with the corresponding edges.

On the upper left we have the patient statistic data compiled by the hospital. The data was originally in a semi-structured format, now it is a TGS.

On the left below we find the demographic data provided by administration office as hierarchical data and now transformed into a TGS.

The target TGS on the right was transformed from 3 relations, which can still be recognized from the target graph.

Before we go to the matching and mapping task 3, I like to present some mapping patterns that reoccur typically during matching and mapping steps.



Now we present a series of typical mappings that arise during schema integration.

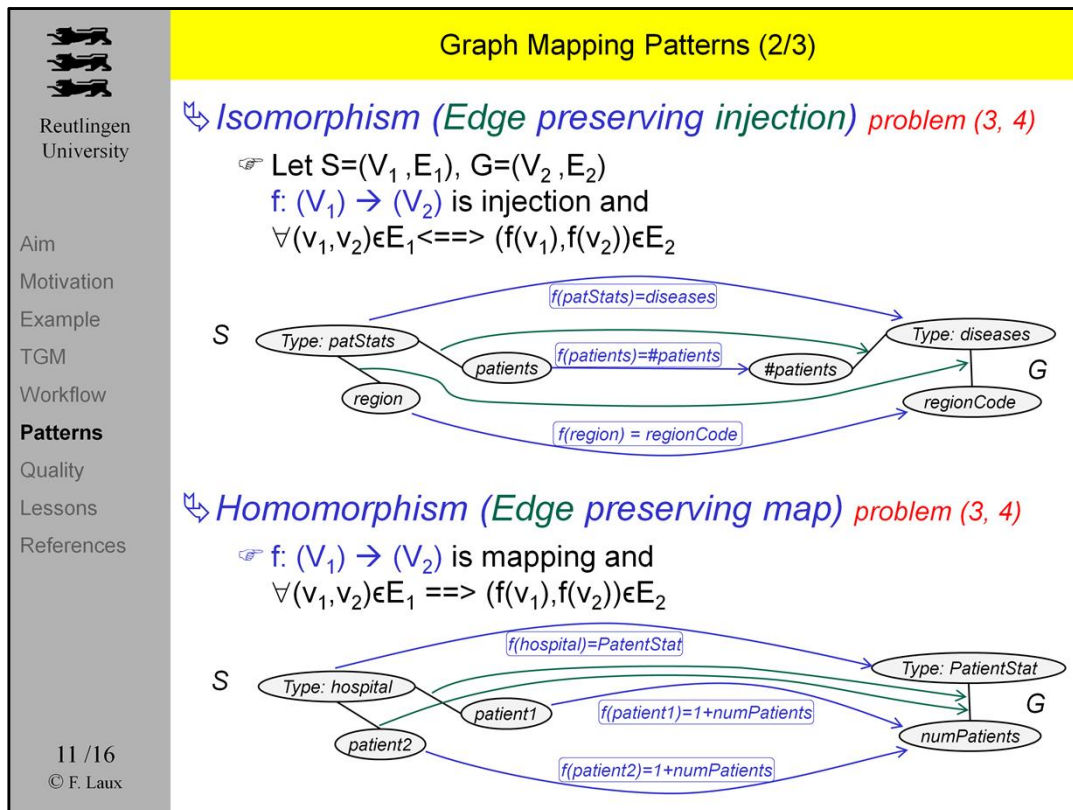
The presented patterns also refer to the problems mentioned for the running example.

The MATCH example refers to problem 1 (identify matching nodes) and problem 3 (preserving semantics)

A match is essentially a 1:1 mapping of 2 data nodes. The matched nodes should have compatible semantics.

A MERGE unions two or more set of nodes. If two nodes have conflicting data a conflict resolution is necessary.

Different data granularity can be aligned by the mapping step.




A graph isomorphism is a edge preserving bijection, that is, a 1:1 mapping.

An isomorphism matches nodes **and** edges.

It helps with problem 3 & 4 , which care about structure and semantics preservation.

A graph homomorphism is similar, it preserves edge types, but allows that multiple nodes are mapped to the same target node.



Reutlingen
University

Aim

Motivation

Example

TGM

Workflow

Patterns

Quality

Lessons

References

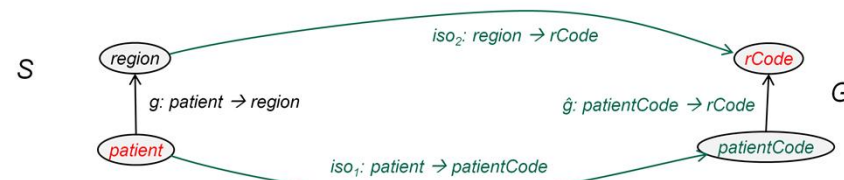
12 / 16

© F. Laux

Graph Mapping Patterns (3/3)

↪ **Commutative Mapping** *refers to problem (5)*

- ☞ A function chain is called **commutative** if and only if $f_2 \circ f_1 = f_1 \circ f_2$, i.e. $f_2(f_1(x)) = f_1(f_2(x)) \forall x \in \text{dom}(f_1)$
- ☞ Example: $g \circ \text{id} = \text{id} \circ g$ (and more general $\hat{g} \circ \text{iso}_1 = \text{iso}_2 \circ g$)



- ☞ For a consistent mapping from **patient** to **rCode** it is irrelevant if the projection g to **region** is done first or the isomorphic mapping iso_1 to **patientCode**.

↪ **Desirable Mappings**

- ☞ Projection π , Homomorphism hom , and Isomorphism iso are good candidates for commutative mappings.
(e.g. $\pi \circ \text{iso} = \text{iso} \circ \pi$)

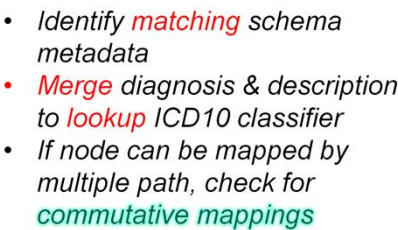
When defining mappings between two schemata special care has to be taken if a target node can be reached via different paths.

This happens for example when in the source schema two data items are related and both items are mapped to the same target node.

In this case we can use the Merge pattern to resolve the conflict.

But if we preserve edges like in our examples we should have mappings that commute.

If special mappings are used like projection, homomorphism, or isomorphism then chances are good that we have commutativity.




The graph of the population table is suppressed to not overload the picture.

The merge operation is next and matches the diagnosis with the ICD description and connects both to lookup the ICD code.

This is highlighted with a green glow.

Both mappings commute because we have an isomorphism and a projection mapping in both paths.

 <p>Reutlingen University</p> <p>Aim Motivation Example TGM Workflow Patterns Quality Lessons References</p> <p>14 / 16 © F. Laux</p>	Quality Control
	<p>↳ <i>Visual inspection of the matching and mapping</i></p> <ul style="list-style-type: none"> ☞ Avoid 1-to-n mappings if possible ☞ Add conflict rules for merge mappings if necessary ☞ Use theorem of Hall for completeness and coverage checking <p>↳ <i>Check formal consistency.</i></p> <ul style="list-style-type: none"> ☞ If a target node can be reached by more than one path, make sure that the mappings are commutative. ☞ If the mapping is an isomorphism, the mapping is lossless. ☞ When the mapping is an aggregation, than the mapping should be a homomorphism. <p>↳ <i>Quality criterion for alternative mappings</i></p> <ul style="list-style-type: none"> ☞ 3 points for 1-1 mappings, 2 points for n-1, and 1 to 3 points for 1-n mappings depending on reliability of the splitting ☞ Prefer mappings with higher score

In the 4th task of the workflow we **check** the integration process **for completeness and validate the integration schema** for formal consistency.

The visual inspections include the following:

Avoid 1-to-n mappings because they indicate that there is a granularity mismatch which can reduce the mapping quality.

Add conflict rules for merge mappings if the sources don't have compatible semantics.


and: **Use the theorem of Hall for completeness and coverage checking.**

Hall's theorem is also known as "marriage theorem because it states the conditions under which all nodes can be matched, that is, a perfect match exists.

If no perfect match exist, a merge conflict can arise and a conflict resolution is necessary.

Check the integration mapping for formal consistency: for instance, look after isomorphism, aggregation and commutative mappings.

In case of alternative mappings use the quality criterion giving 3 points for 1-1 mappings, 2 points for n-1, and 1 to 3 points for 1-n mappings depending on reliability of the splitting. Then, chose the path with the highest score.

 Reutlingen University Aim Motivation Example TGM Workflow Patterns Quality Lessons References 15 / 16 © F. Laux	<div>Lessons learned</div> <ul style="list-style-type: none">↪ <i>Use the TGM as intermediate “supermodel” for source and target schemata</i>↪ <i>Visually inspect the matching and mapping</i>↪ <i>Some graph mapping theorems allow formal and automated quality checking</i><ul style="list-style-type: none">☞ Theorem of Hall: coverage/completeness check☞ Commutative mappings: consistency checks☞ Hypergraph links for merge mappings need conflict rules and splitting mapping need distribution description
---	--


We have presented the TGM and recommend it as intermediate “supermodel” for source and target schemata.

The mapping and matching requires human inspection and quality control.

Some graph mapping theorems allow formal and automated quality checks, for instance, the theorem of Hall.

Check the consistency of commutative mappings and choose the one with the highest quality score.

Hypergraph links for merge mappings need conflict rules and splitting mappings require a detailed distribution description.

	Selected References
Reutlingen University Aim Motivation Example TGM Workflow Patterns Quality Lessons References 16 / 16 © F. Laux	<p>↗ P. Bernstein and L. Haas, "Information integration in the enterprise", <i>Communications of the ACM</i> 51(9), pp. 72--79, 2008.</p> <ul style="list-style-type: none"> ☞ Visual Mapping tool, semi-automatic matching with human validation, subtle matching problems, and complex transformation. <p>↗ P. Bernstein and S. Melnik, "Model Management 2.0: Manipulating Richer Mappings", <i>Proceedings of the ACM SIGMOD</i>, pp. 1--12, 2007.</p> <ul style="list-style-type: none"> ☞ Model management is a generic approach to precisely engineered mappings and schema evolution <p>↗ F. Laux, "The Typed Graph Model", <i>DBKDA 2020</i>, pp. 13--19, ISBN: 978-1-61208-790-0.</p> <ul style="list-style-type: none"> ☞ formally defines the TGM, comparison with other models by example. <p>↗ M. Lenzerini, "Data Integration: A Theoretical Perspective", <i>ACM PODS 2002</i>, pp. 233-246, 2002</p> <ul style="list-style-type: none"> ☞ Gives a compact overview of different approaches and discusses its consequences <p>↗ P. Atzeni, P. Cappellari, R. Torlone, Ph. Bernstein, and G. Gianforme, "Model-independent schema translation", <i>VLDB Journal</i> 17(6), pp. 1347--1370, 2008</p> <ul style="list-style-type: none"> ☞ Propose a semantic super-model that covers all prevalent models based on the generic semantic model of Hull and King (1987)

This is a short selection of the most important papers used for our work.

Bernstein and Haas present a visual mapping tool with human validation and describe problems for semi-automatic matching.

Bernstein and Melnik present a generic approach to precisely engineered mappings and schema evolution.

Laux introduces the TGM and compares it with other popular data models.

Lenzerini gives a compact overview of different integration approaches and discusses its consequences.

Atzeni and his colleagues propose a semantic super-model that covers all prevalent models based on the generic semantic model of Hull and King.