



A Philosophy of Modeling

Lars Rönnbäck



up
to
change



**No two objects can have all the
same properties.**

Gottfried Wilhelm Leibniz



Everything flows.

Heraclitus



All things are momentary.

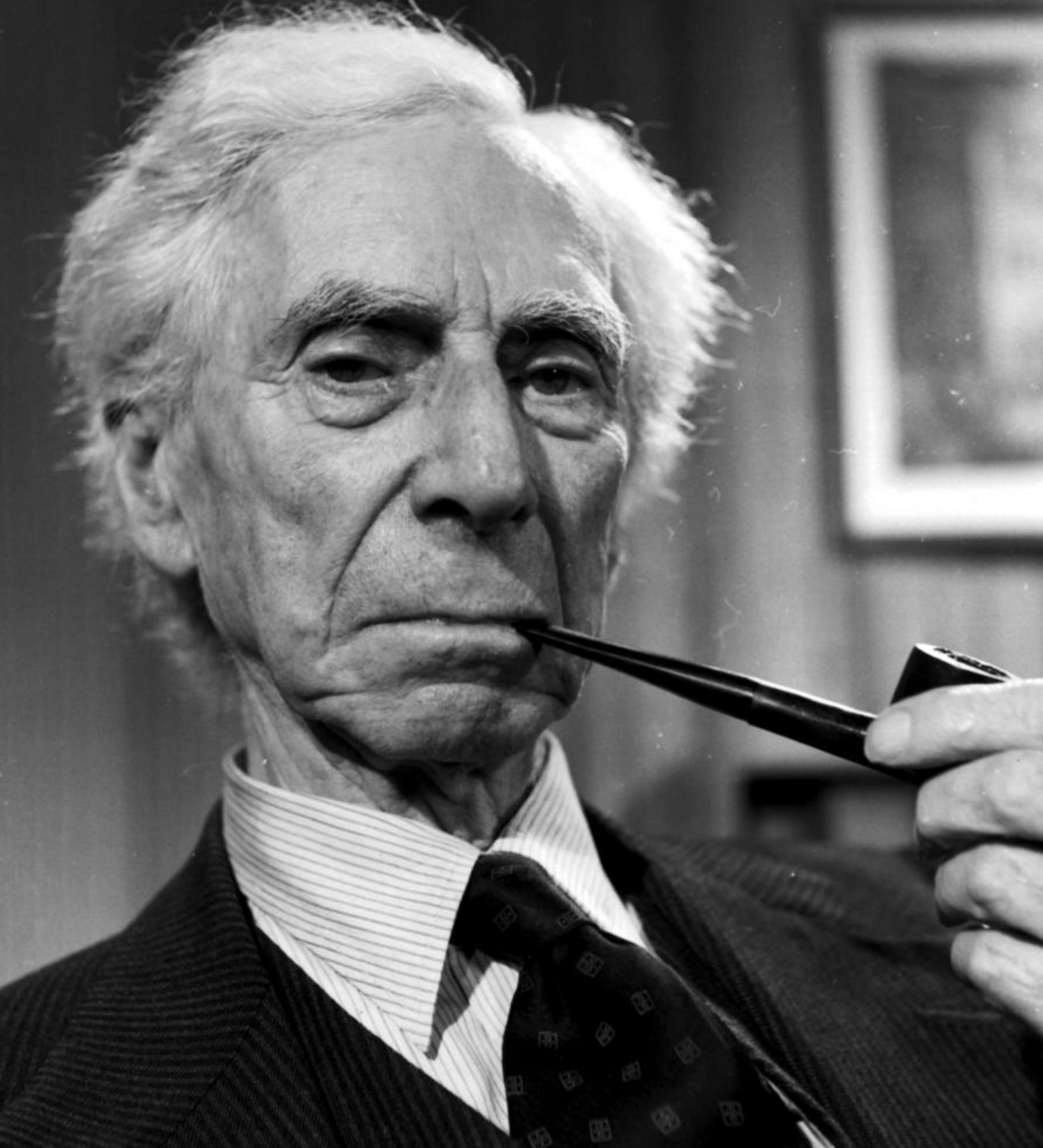
Vasubandhu

The Buddhist Doctrine of Momentariness



The reality we can put into
words is never reality itself.

Werner Heisenberg



**Some things are more
nearly certain than others.**

Bertrand Russell



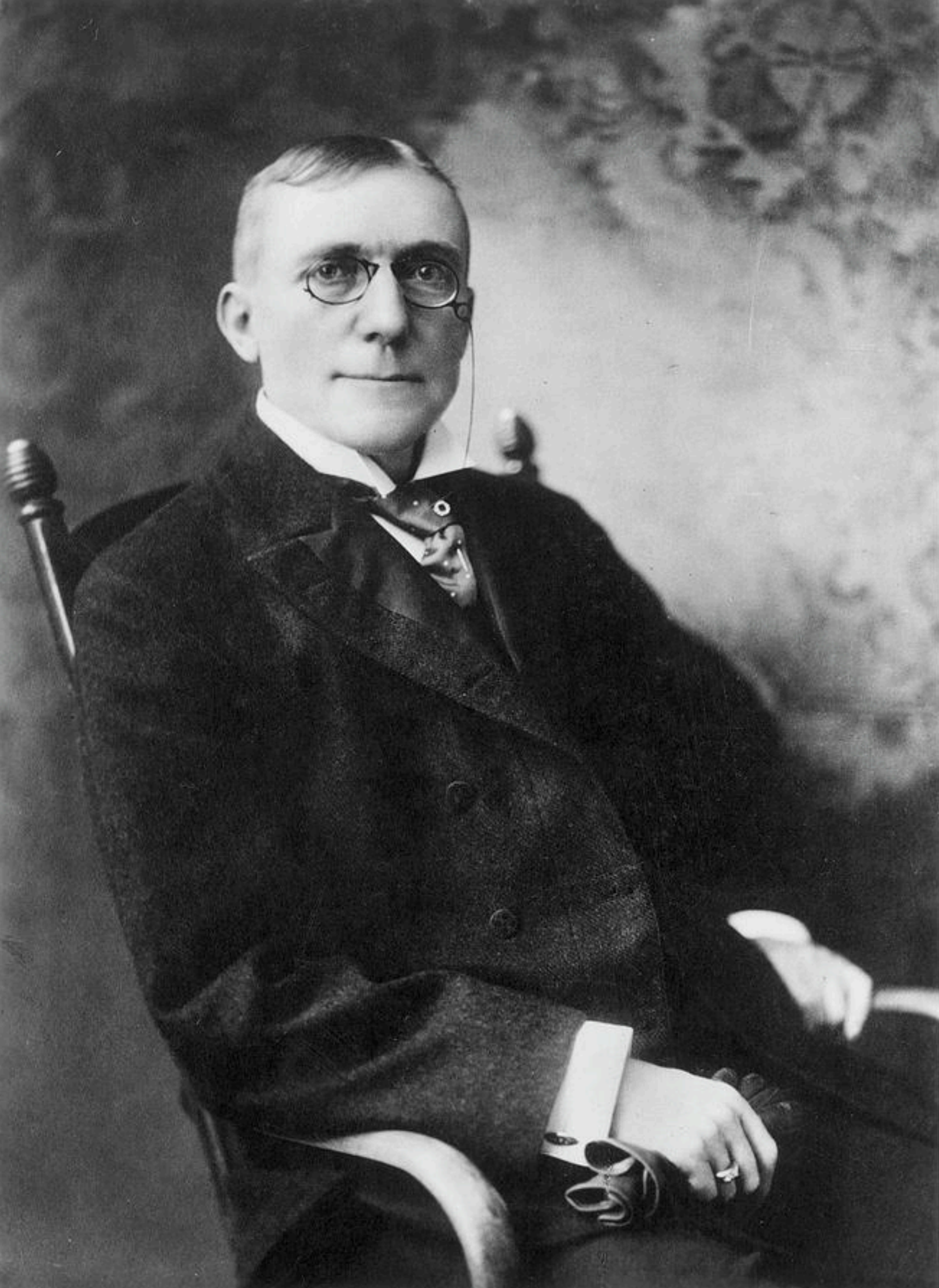
Our subjectivity is so
completely our own.

Spike Jonze



**One should always aim
at being interesting,
rather than exact.**

Voltaire



If it quacks like a duck...

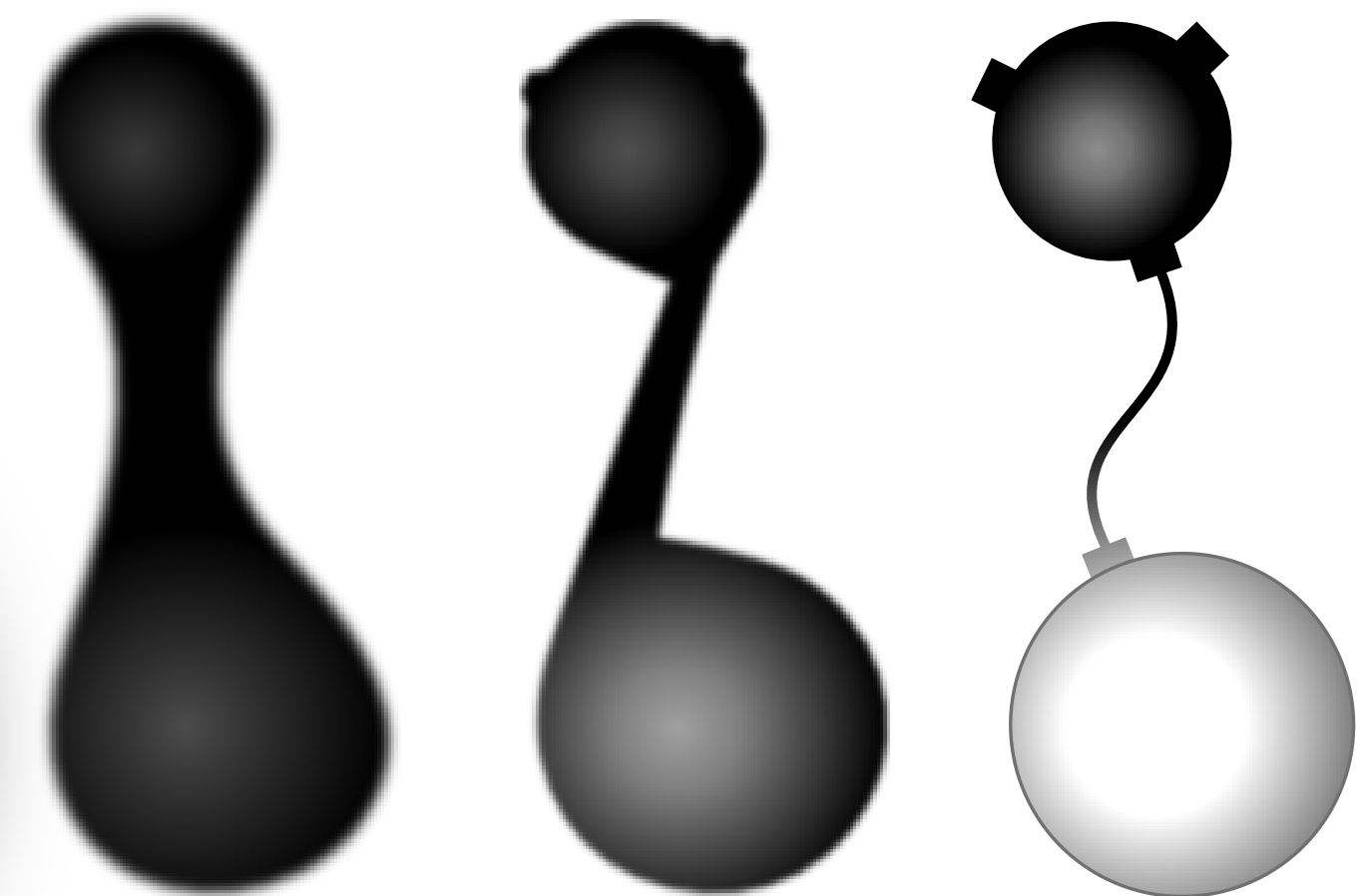
James Whitcomb Riley



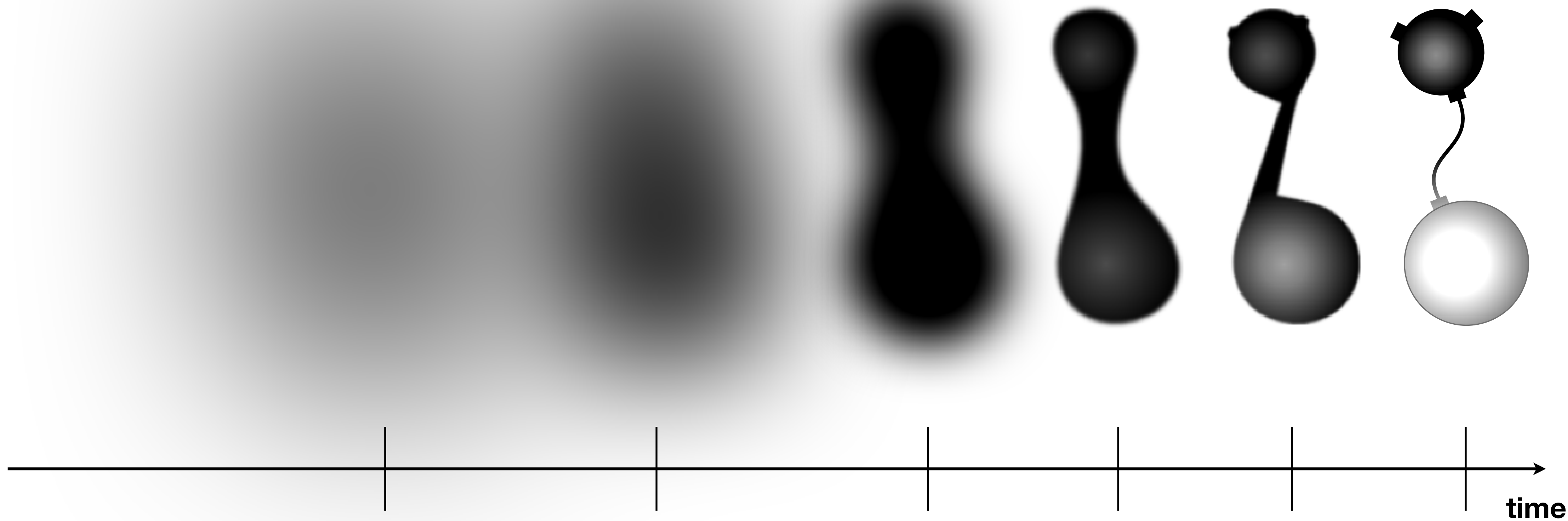
Same same, but different.

Thai salesperson

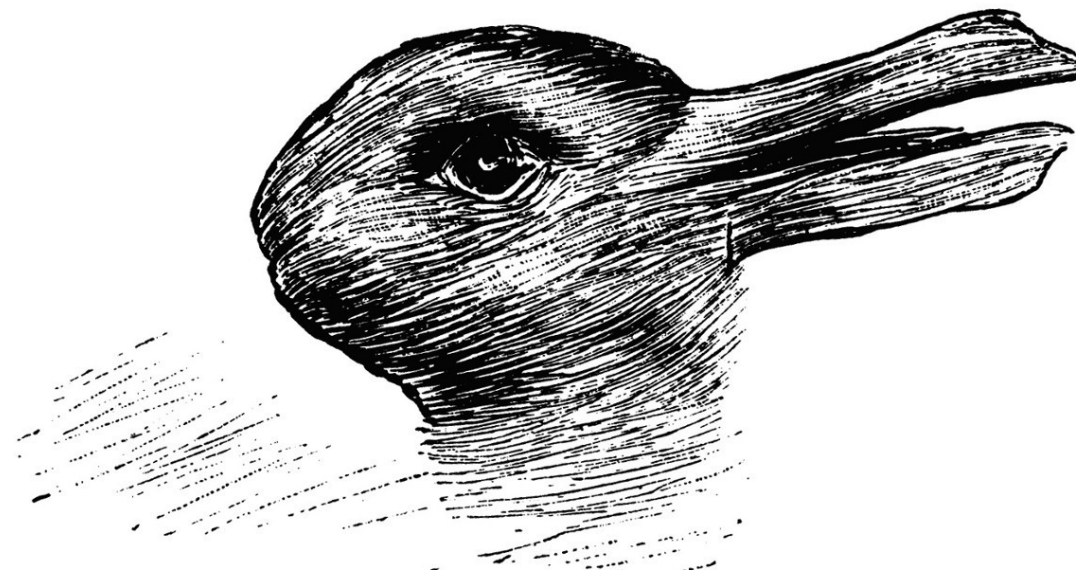
When is a thing *a thing*?



When is a thing *a thing*?



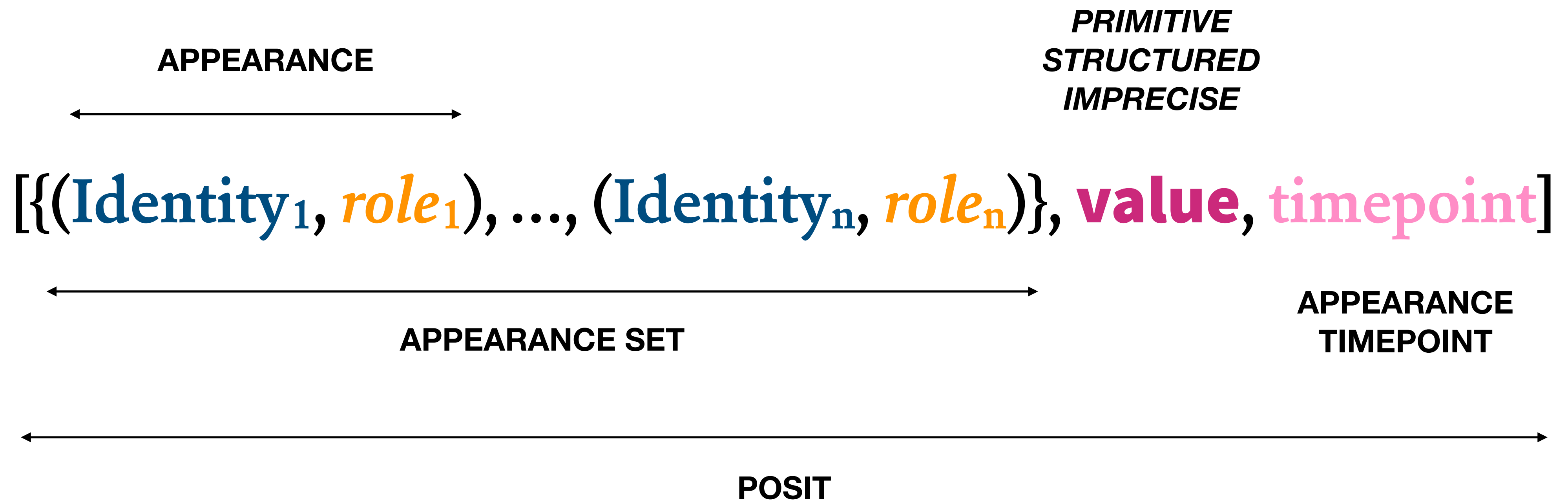
- Things are important
- Their properties are important
- Their classification is ephemeral
- Values are always imprecise
- Changes can be captured through sampling
- Only identities are immutable
- We record statements about reality, and we do not record reality itself
- The certainty of true reality cannot be captured
- Opinions may differ and may be revised
- Keep the number of things that have to be agreed upon small



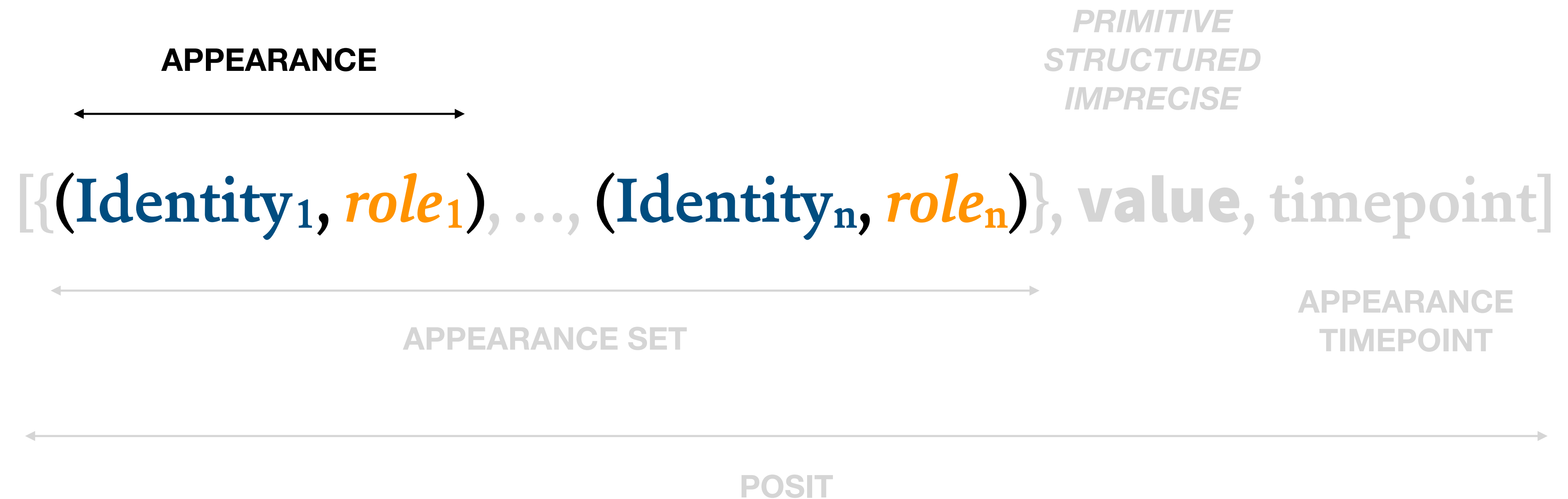
Transitional Modeling

Lars Rönnbäck

$[\{ (,), \dots, (,) \}, ,]$



The least we must agree upon are appearances



...which lets us disagree about everything else

[{(42, *husband*), (43, *wife*)}, **married**, 2004-06-19]

[{(42, *name*)}, **Lars Samuelsson**, 1972-08-20]
[(42, *hair color*)}, **gray**, 2022-02-22]

[{(42, *name*)}, **Lars Samuelsson**, 1972-08-20]

[{(42, *name*)}, **Lars Rönnbäck**, 2005-03-30]

[{(42, *hair color*)}, **brown**, 1973-02-13]

[{(42, *hair color*)}, **gray**, 2022-02-22]

VALUES MAY CHANGE OVER TIME (*APPEARANCE TIMELINE*)

DATA	[{(42, <i>name</i>)}, Lars Rönnbäck , 2005-03-30]		
DATA	[{(42, <i>hair color</i>)}, gray , 2022-02-22]		
DATA	[{(4711, <i>name</i>)}, Person , 2019-10-20]		
PERIDATA	[{(42, <i>thing</i>), (4711, <i>class</i>)}, active , 1972-08-20]		
	<table><tbody><tr><td>RESERVED ROLE</td><td>RESERVED ROLE</td></tr></tbody></table>	RESERVED ROLE	RESERVED ROLE
RESERVED ROLE	RESERVED ROLE		

POSIT
IDENTITY

555 ← [{(42, *name*)}, **Lars Rönnbäck**, 1972-08-20]

556 ← [{(42, *hair color*)}, **gray**, 2022-02-22]

557 ← [{(42, *husband*), (43, *wife*)}, **married**, 2004-06-19]

POSIT
IDENTITY

555 ← [{(42, *name*)}, **Lars Rönnbäck**, 1972-08-20]

556 ← [{(42, *hair color*)}, **gray**, 2022-02-22]

557 ← [{(42, *husband*), (43, *wife*)}, **married**, 2004-06-19]

CERTAINTY ASSERTION
TIMEPOINT

[{(555, *posit*), (42, *ascertains*) }, **90%**, 2019-10-20]



ASSERTION

555 ← [{(42, *name*)}, **Lars Rönnbäck**, 1972-08-20]

556 ← [{(42, *hair color*)}, **gray**, 2022-02-22]

557 ← [{(42, *husband*), (43, *wife*)}, **married**, 2004-06-19]

[{(555, *posit*), (42, *ascertains*)}, **90%**, 2019-10-20]

[{(556, *posit*), (42, *ascertains*)}, **-80%**, 2019-10-20]

[{(557, *posit*), (42, *ascertains*)}, **0%**, 2019-10-21]

COMPLETE
UNCERTAINTY

557 ← [{(42, *husband*), (43, *wife*)}, **married**, 2004-06-19]

558 ← [{(42, *husband*), (43, *wife*)}, **married**, 2005-06-19]

CERTAINTY MAY CHANGE OVER TIME (ASSERTION TIMELINE)

[{(557, *posit*), (42, *ascertains*)}, **90%**, 2019-10-20]

[{(557, *posit*), (43, *ascertains*)}, **-100%**, 2019-10-20]

[{(557, *posit*), (42, *ascertains*)}, **0%**, 2019-10-21]

[{(558, *posit*), (42, *ascertains*)}, **100%**, 2019-10-21]

[{(558, *posit*), (43, *ascertains*)}, **100%**, 2019-10-21]

555 ← [{(42, *name*)}, **Lars Rönnbäck**, 1972-08-20]

556 ← [{(42, *hair color*)}, **gray**, 2022-02-22]

557 ← [{(42, *husband*), (43, *wife*)}, **married**, 2004-06-19]

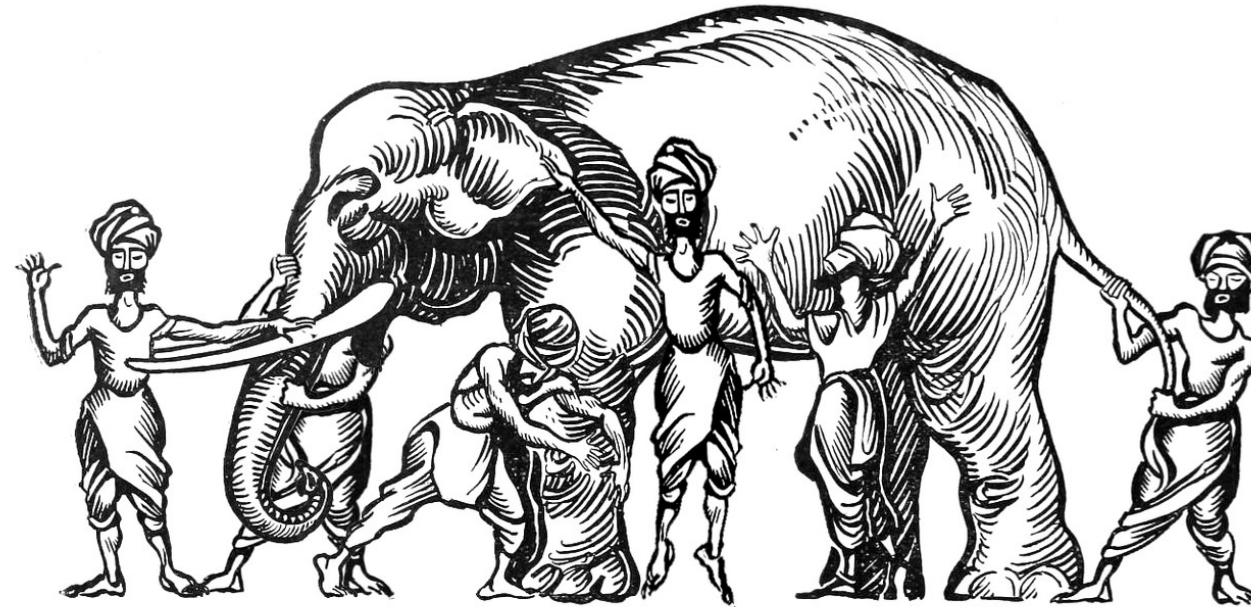
METADATA

[{(555, *posit*), (2001, *job*)}, **success**, 2019-10-20]

[{(2001, *started at*)}, **2019-10-20 10:15**, 2019-10-20]

[{(2001, *user*)}, **SQLAgent**, 2019-10-20]

- From here we could go on and also define:
 - Constraints (and cardinality) as posits
 - Identifiers (natural keys) as posits
 - Structures (such as ensembles) as posits
 - Inheritance, Trustworthiness, Consensus, Contradictions, and so on...



Schemaful Databases

How to model without classes

Lars Rönnbäck

beard
color

name

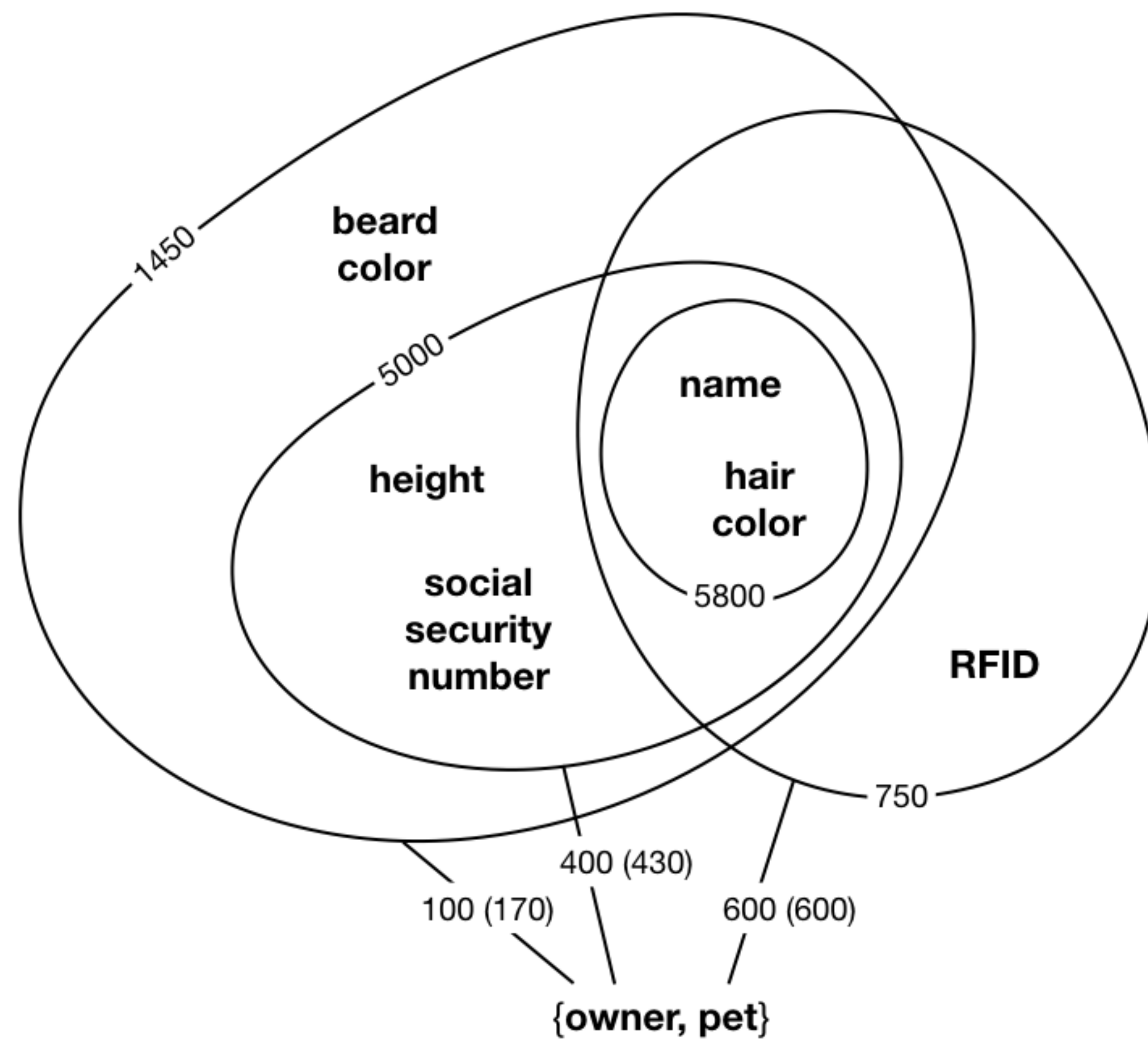
height

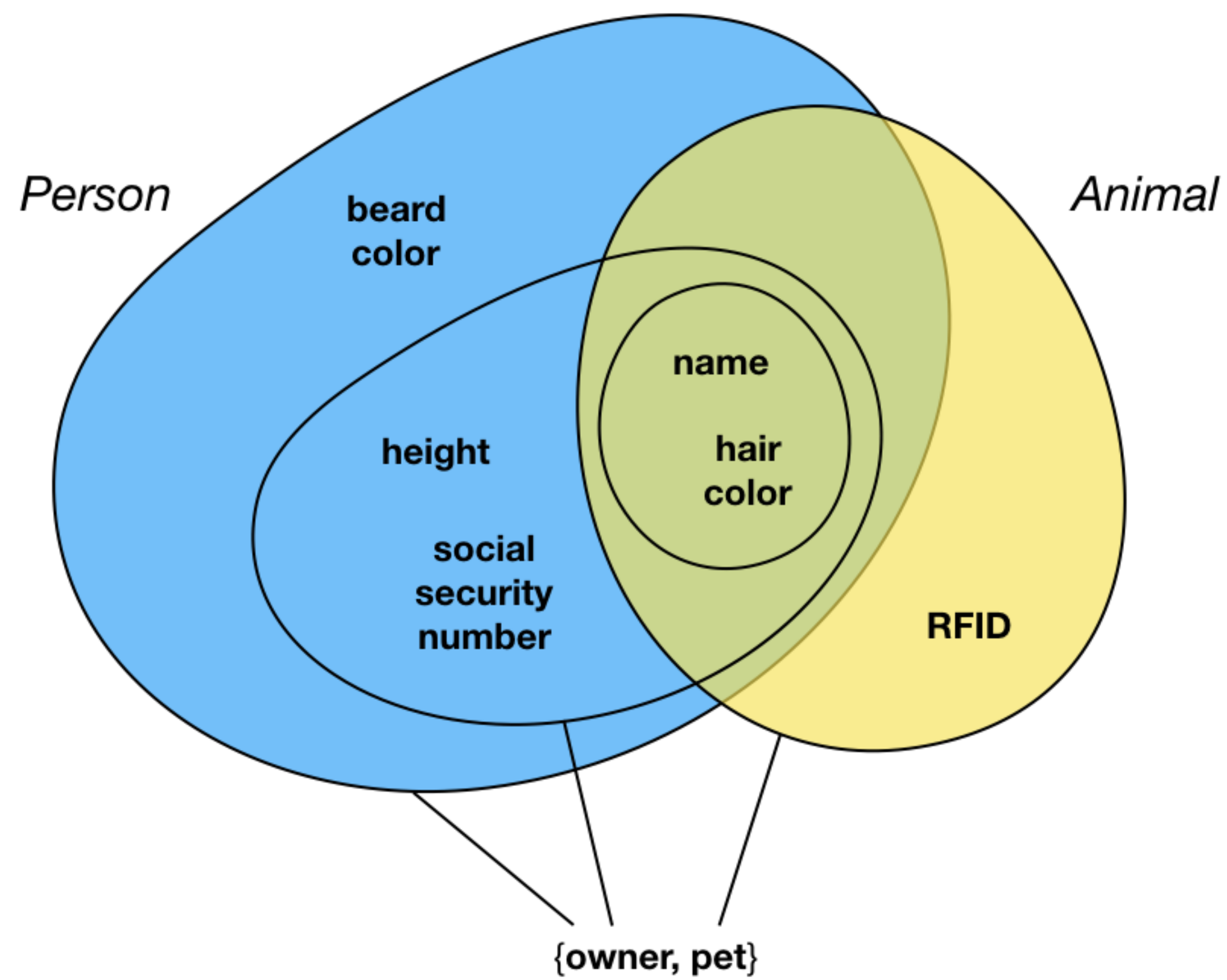
hair
color

social
security
number

RFID

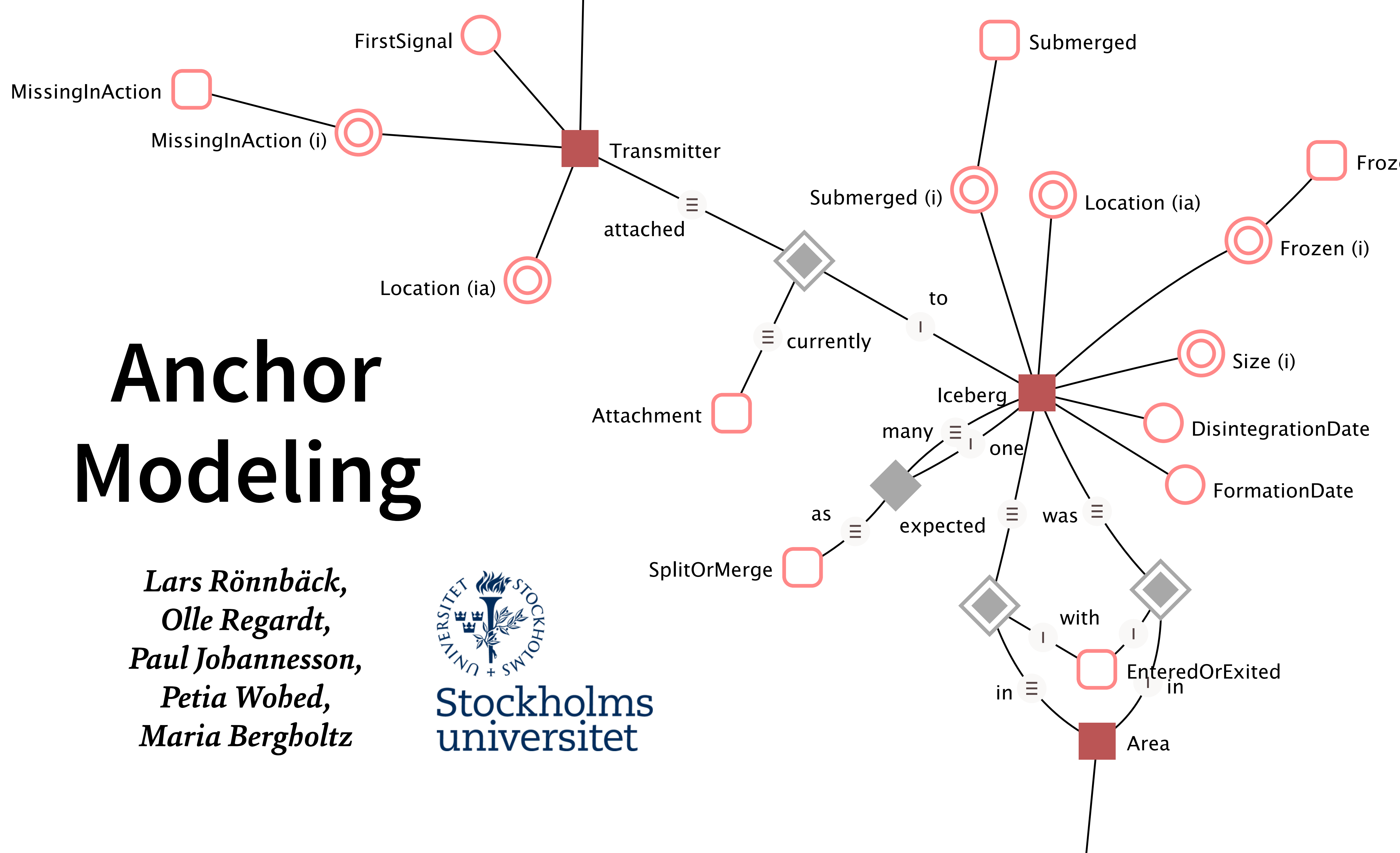
owner, pet





Anchor Modeling

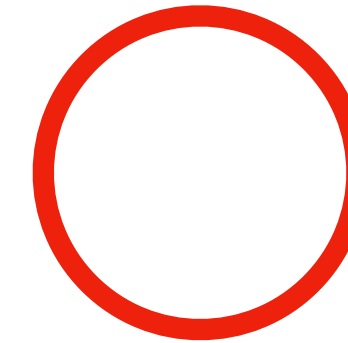
*Lars Rönnbäck,
Olle Regardt,
Paul Johannesson,
Petia Wohed,
Maria Bergholtz*



DATA [(42, name), Lars Rönnbäck, 1972-08-20]
DATA [(42, hair color), gray, 2022-02-22]
DATA [(4711, name), Person, 2019-10-20]
PERIDATA [(42, thing), (4711, class), active, 1972-08-20]



An *anchor* named PE_Person holds the identities for things of the Person class.



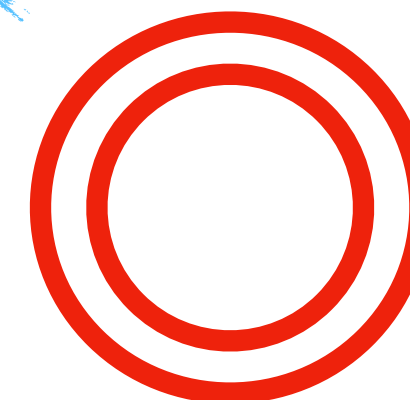
A static attribute named
PE_NAM_Person_Name holds a reference
to an **identity** and a **primitive value**.

DATA [(42, name), Lars Rönnbäck, 1972-08-20]

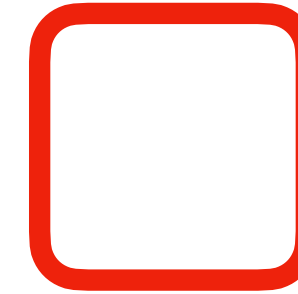
DATA [(42, hair color), gray, 2022-02-22]

DATA [(4711, name), Person, 2019-10-20]

PERIDATA [(42, thing), (4711, class), active, 1972-08-20]



A historized attribute named
PE_HAC_Person_HairColor holds a reference
to an **identity**, a **primitive value**, and a
time point since when it came into effect.



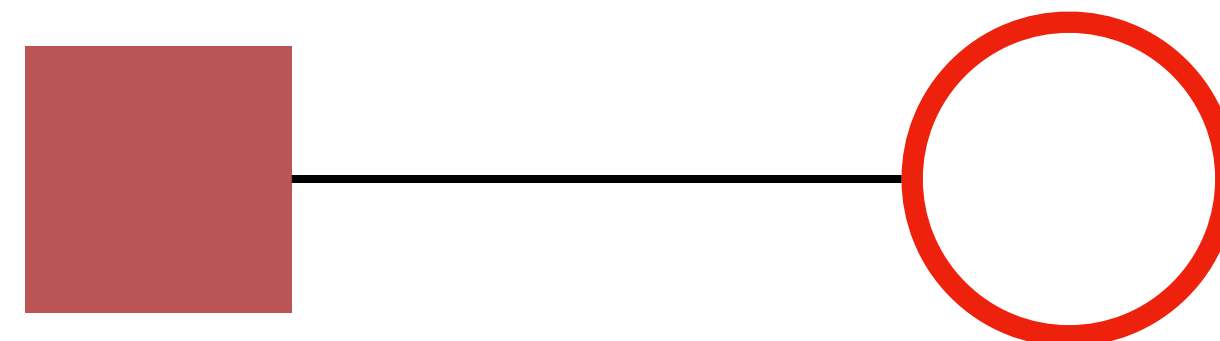
A *knot* named COL_Color holds an enumeration of **primitive values** along with their own **identites**.

DATA [{(42, *name*)}, **Lars Rönnbäck**, 1972-08-20]

DATA [{(42, *hair color*)}, **gray**, 2022-02-22]

DATA [{(4711, *name*)}, **Person**, 2019-10-20]

PERIDATA [{(42, *thing*), (4711, *class*)}, **active**, 1972-08-20]



Knots are equivalent to an anchor with a single static attribute.

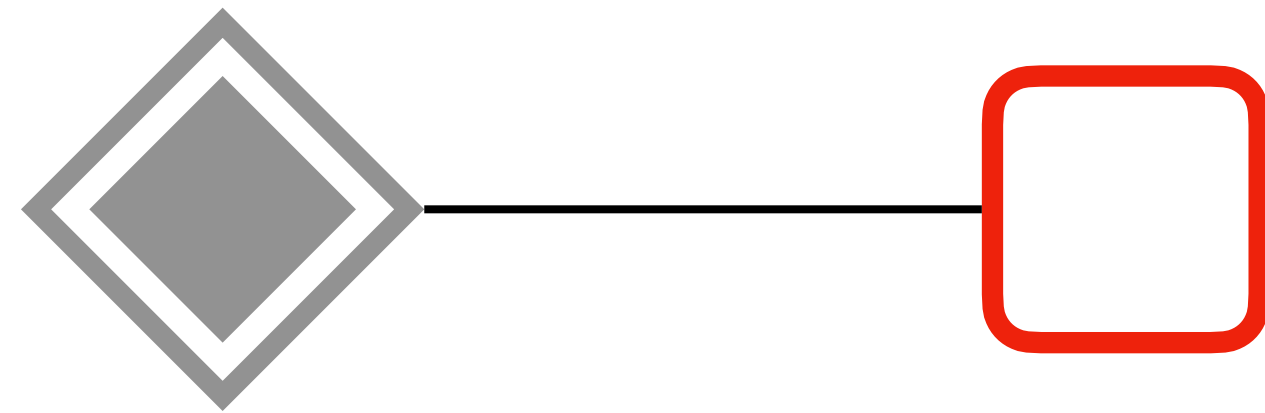


A *static tie* named PE_lecturer_IN_invoice holds references to **identities** in adjoined anchors.

[[{(42, *lecturer*), (911, *invoice*)}, **created**, 2019-10-20]
[[{(42, *husband*), (43, *wife*)}, **married**, 2004-06-19]

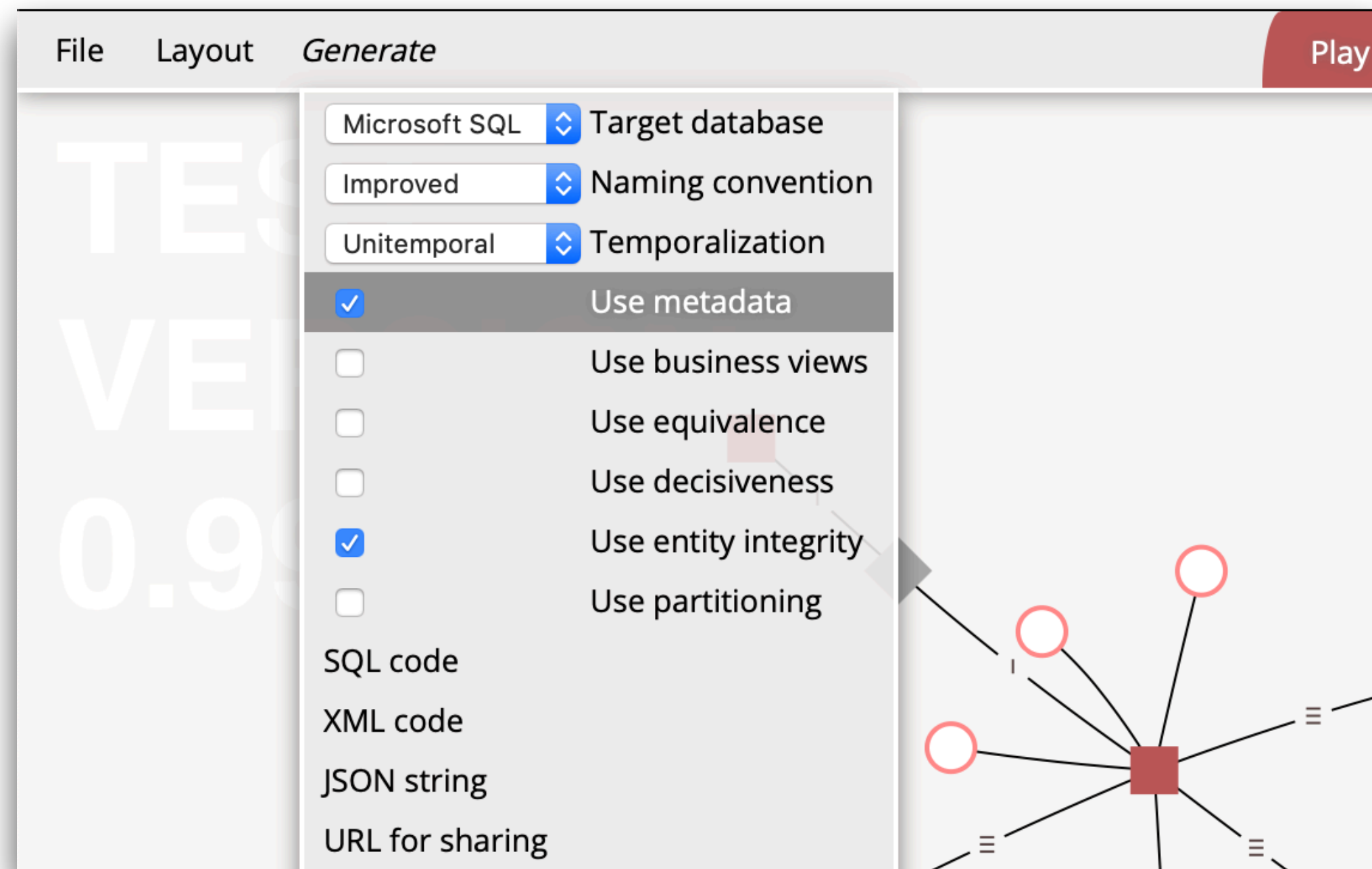


A *historized tie* named PE_husband_PE_wife holds references to **identities** in adjoined anchors, and a **time point** indicating since when the relationship has been in effect.



A *knotted historized tie* named PE_husband_PE_wife_STA_currently holds references to **identities** in adjoined anchors, a **time point** indicating since when the relationship has been in effect, and the **identity** of an enumerated value from a knot.

[(42, husband), (43, wife)], married, 2004-06-19]



A *knotted historized tie* named PE_husband_PE_wife_STA_currently holds references to **identities** in adjoined anchors, a **time point** indicating since when the relationship has been in effect, and the **identity** of an enumerated value from a knot.

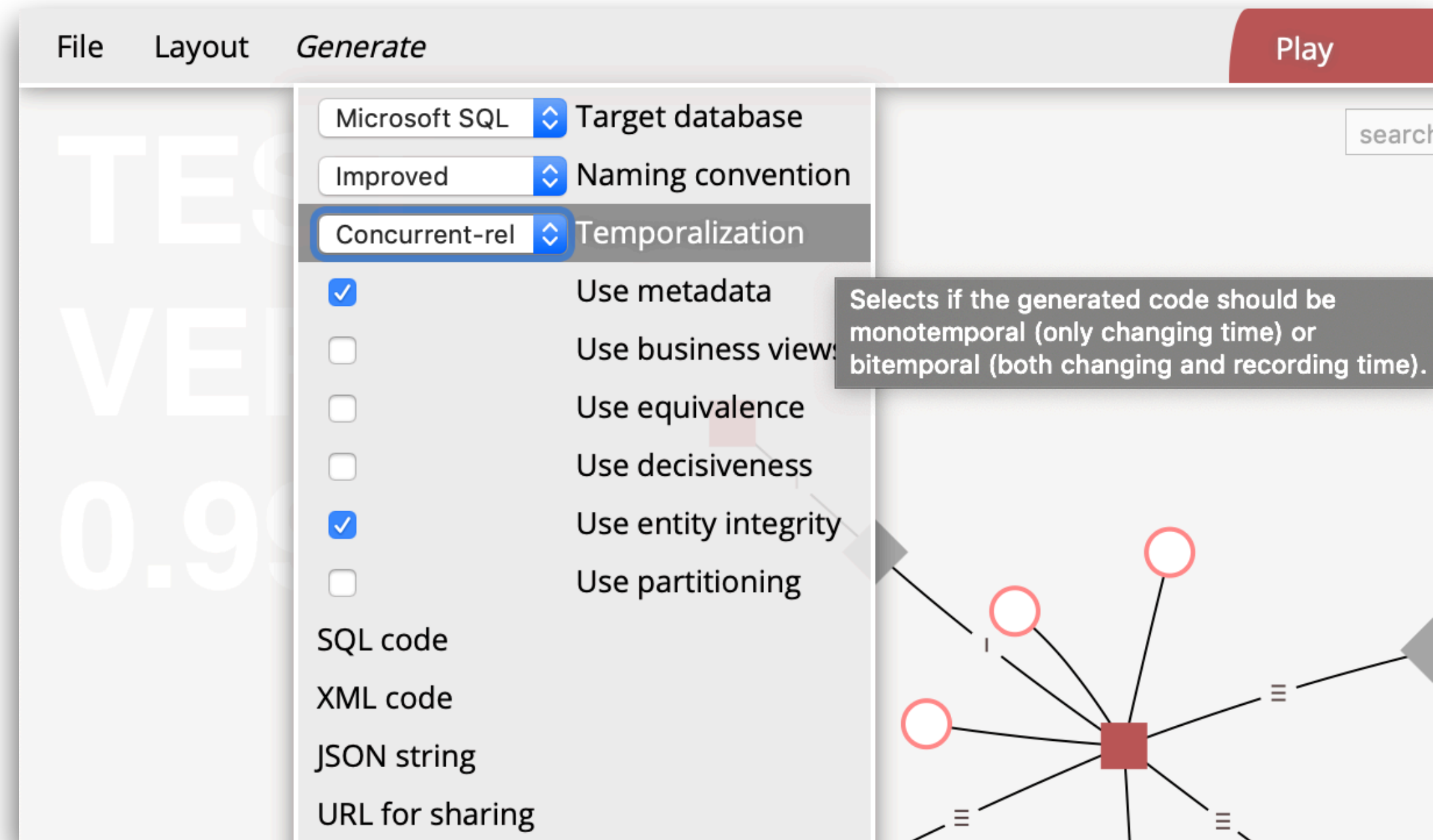
With metadata it also holds a reference to a metadata **identity**.

555 ← [(42, *name*), **Lars Rönnbäck**, 1972-08-20]

[(555, *posit*), (2001, *job*), **success**, 2019-10-20]

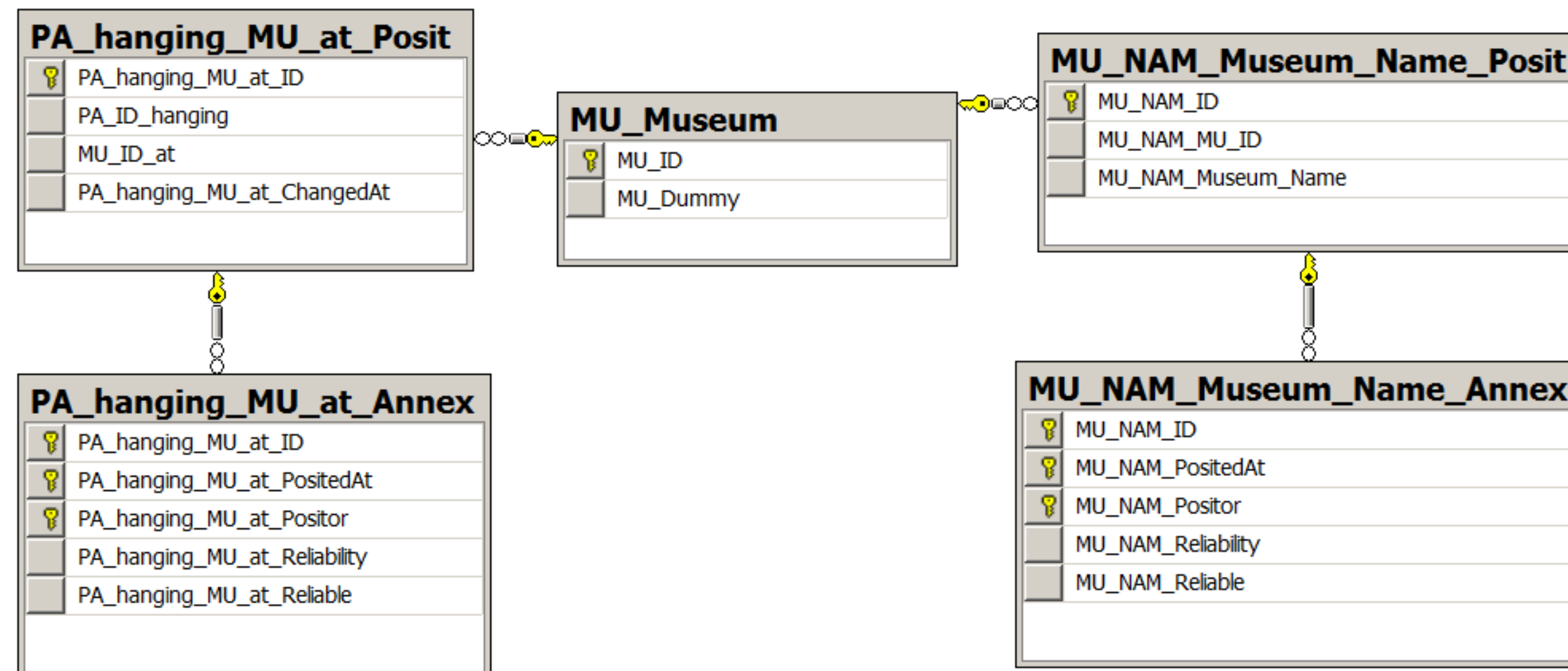
[(2001, *started at*), **2019-10-20 10:15**, 2019-10-20]

[(2001, *user*), **SQLAgent**, 2019-10-20]



In *uni-temporal* Anchor modeling, only posits are stored.

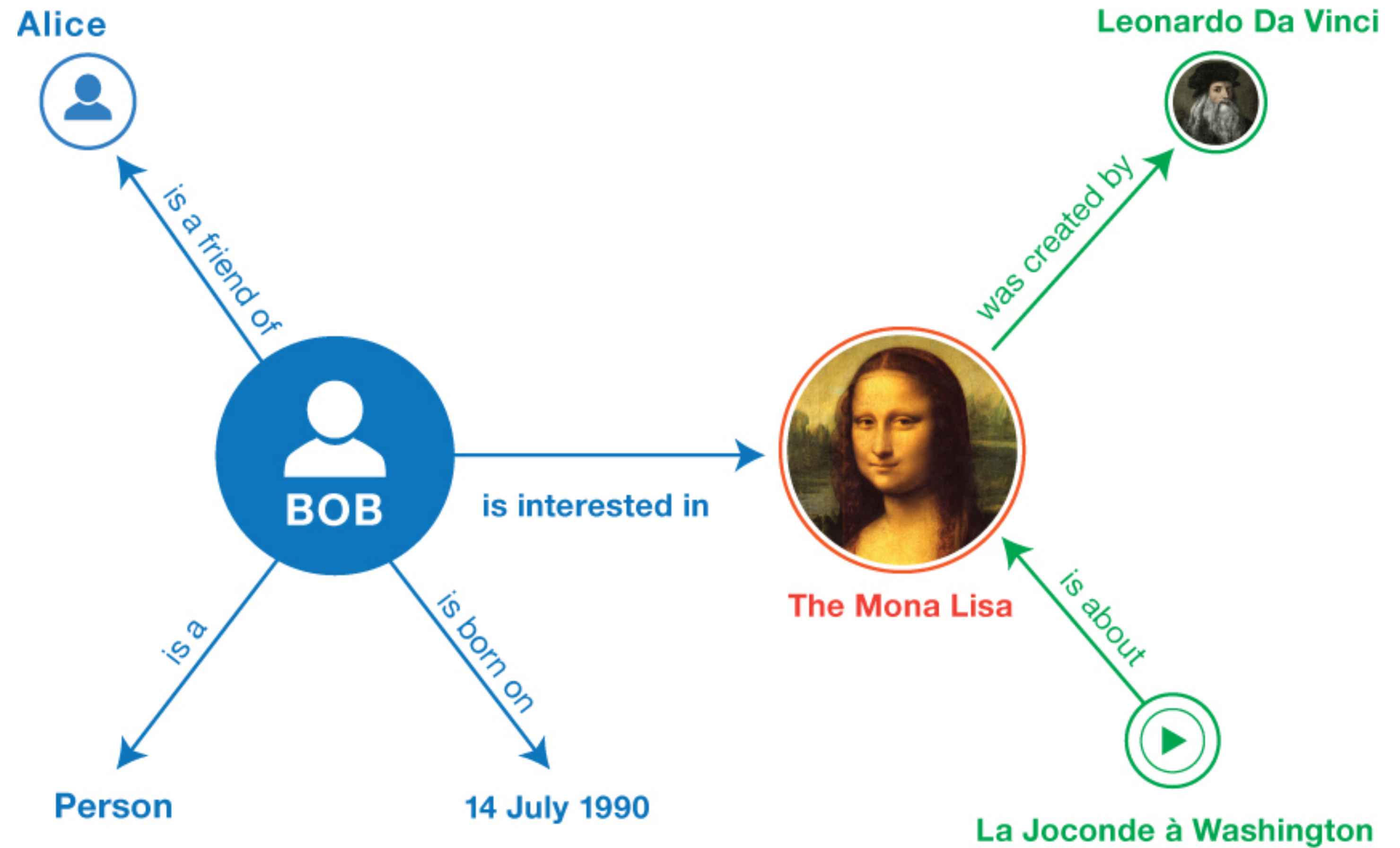
In *concurrent-reliance-temporal* Anchor modeling an Annex table is added, in which assertions are stored.





Semantic Triples

*World Wide Web
Consortium*





<Bob> <is a> <Person>

<Bob> <is a friend of> <Alice>

<Bob> <is born on> <the 4th of July 1990>

<http://example.name#BS12> <known as> <Bob>

<http://example.name#AW8> <known as> <Alice>



<Bob> <is a> <Person>

<Bob> <is a friend of> <Alice>

<Bob> <is born on> <the 4th of July 1990>

<http://example.name#BS12> <known as> <Bob>

<http://example.name#AW8> <known as> <Alice>

[{(http://example.name#BS12, *thing*), (http://xmlns.com/foaf/0.1/Person, *class*)},
active, 1990-07-04]

[{(http://example.name#BS12, *is a friend*), (http://example.name#AW8, *of*)},
active, 2014]

[{(http://example.name#BS12, *is born on*)}, **the 4th of July 1990**, 1990-07-04]

[{(http://example.name#BS12, *known as*)}, **Bob**, 1990-07-04]

[{(http://example.name#AW8, *known as*)}, **Alice**, 1992-10-11]

<Person> <type> <Class>
<is a friend of> <type> <Property>
<is a friend of> <domain> <Person>
<is a friend of> <range> <Person>





<Person> <type> <Class>
<is a friend of> <type> <Property>
<is a friend of> <domain> <Person>
<is a friend of> <range> <Person>

[{(http://xmlns.com/foaf/0.1/Person, *type*)}, **Class**, 1998]

[{(http://xmlns.com/foaf/0.1/Dog, *type*)}, **Class**, 1998]

[{(http://xmlns.com/foaf/0.1/Person, *is a friend*),
(http://xmlns.com/foaf/0.1/Person, *of*)},
property, 1998]

CONSTRAINTS

[{(http://xmlns.com/foaf/0.1/Dog, *is a friend*),
(http://xmlns.com/foaf/0.1/Person, *of*)},
property, 1998]



<Person> <type> <Class>
<is a friend of> <type> <Property>
<is a friend of> <domain> <Person>
<is a friend of> <range> <Person>
<is a good friend of> <subPropertyOf> <is a friend of>

901 ← [{(http://xmlns.com/foaf/0.1/Person, *is a good friend*),
(http://xmlns.com/foaf/0.1/Person, *of*)},
property, 1998]

902 ← [{(http://xmlns.com/foaf/0.1/Person, *is a friend*),
(http://xmlns.com/foaf/0.1/Person, *of*)},
property, 1998]

[{(901, *constraint is*), (902, *constraint of*)}, **sub property**, 2001]

RELATED
CONSTRAINTS



Additional Resources

Lars Rönnbäck



An Introduction to Anchor Modeling

An agile information modeling technique for evolving data environments

Lars Rönnbäck

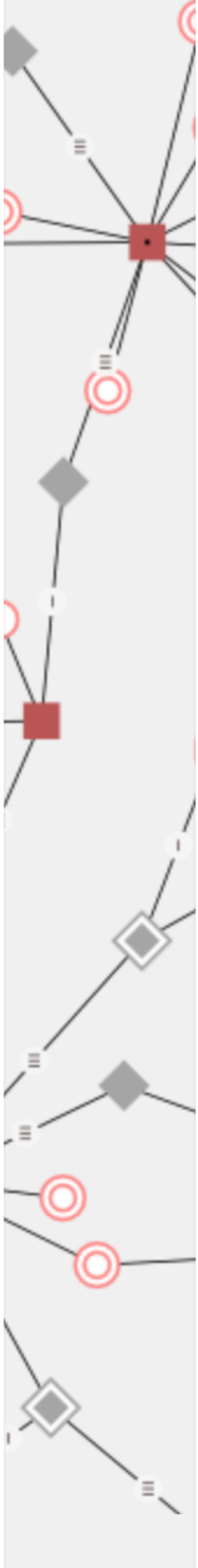
\$299

PAGES

About

Online Modeler

Online Modeler (test version)



Appearance is Everything

In my previous article “[What needs to be agreed upon](#)“, from my series about [#transitional modeling](#), I listed the few things that must be interpreted equally among those sharing information between them. To recall, these were identities, values, roles, and time points. If we do not agree upon these, ambiguities arise, and it is no longer certain that we are talking about the same thing. We used this to create the fundamental construct in transitional modeling; the *posit*, which is a “triple” on the form $[(id^1, role^1), ..., (id^n, role^n)]$, value, time point. A *posit* in position is called a *dereferencing set*, and each *posit* in a *set* is called an *appearance*. An appearance consists of a *posit* and a *set* and they will be the topic of this article.

What is interesting and different from most other articles is that what the identities represent may be subjects, individuals exchanging information in transitional modeling, on the classifications of the things they discuss, or even an identity 42 is thought of as a ‘Living Thing’ by one, a ‘Person’ by a third, a ‘Customer’ by a fourth, and an ‘Animate Object’ by a sixth, a ‘Transaction’ by a seventh, and a ‘Data Point’ by an eighth. This is not to say that these are mutually exclusive, but rather that they are all valid and can be used to describe the same thing. This is the power of transitional modeling; it allows us to create a common language for sharing information between different groups of people, and it allows us to create a common language for sharing information between different groups of people.

MODELING CONFLICTING, UNRELIABLE, AND VARYING INFORMATION

LARS RÖNNBÄCK

FOURTH REVISION — 16 DECEMBER 2018

Most persistent memories in which bodies of information are stored can only provide a view of that information as it currently is, from a single point of view, and with no respect to its reliability. This is a poor reflection of reality, because information changes over time, may have many and possibly disagreeing origins, and is far from often certain. Hereat, this paper introduces a modeling technique that manages conflicting, unreliable, and varying information. In order to do so, the concept of a “single version of the truth” must be abandoned and replaced by an equivocal theory that respects the genuine nature of information. Through such, information can be seen from different and concurrent perspectives, where each statement has been given a reliability ranging from being certain of its truth to being certain of its opposite, and when that reliability or the information itself varies over time, changes are managed non-destructively, making it possible to retrieve everything as it was at any given point in time. As a result, other techniques are, among them third normal form, anchor modeling, and data vault, contained as special cases of the henceforth entitled *transitional modeling*¹.

KEYWORDS
transitional, modeling, information, temporality, concurrency, reliability, variability, language, vagueness, fact

ALL CURRENT information modeling techniques result in lossy implementations, in the sense that they cannot preserve combinations of who said what, when, and how sure they were of what they were saying. Modeling such requirements explicitly using traditional techniques is complex and error-prone, and therefore

Temporal Dimensional Modeling

LARS RÖNNBÄCK^{*}
OLLE REGARDT

the Department of Computer Science, Stockholm University
lars.ronnback@anchormodeling.com

Abstract

One of the prevalent techniques for modeling data warehouses is and has for the last decade been dimensional modeling. As initially defined, it had no constructs for keeping a record of changes and only provided the as-is latest view of available information. Since its introduction and from increasing requirements to record changes, different approaches have been suggested to manage change, mainly in the form of slowly changing dimensions of various types. This paper will show that every existing type of slowly changing dimension may lead to undesired anomalies, either at read or at write time, making them unsuitable for application in performance critical or near real-time data warehouses. Instead, based on current research in temporal database modeling, we introduce temporal dimensions that make facts and dimensions temporally independent, and therefore safer from none of said anomalies. In our research, we also discovered the twine, a new concept that may significantly improve performance when loading dimensions. Code samples, along with query results showing the positive impact of implementing temporal dimensions compared to slowly changing dimensions are also presented.

DIMENSIONAL MODELING · MICROBATCH · DATA WAREHOUSE · SLOWLY CHANGING DIMENSION · HIGH PERFORMANCE · TWINE · TEMPORAL DIMENSION · REAL-TIME ANALYTICS

1. INTRODUCTION

For the last 15 years, two techniques have been dominating when it comes to data warehouse implementations. (Kimball and Ross 2002), with their *dimensional modeling* and (Inmon 2005) with his *normalized model* for data warehouses. Initially, neither had constructs or guidelines for managing data that changes over time, but as such requirements became more common, those features were retrofitted onto these technologies. This paper focuses on the concept of slowly changing dimensions (Kimball 2008; Ross 2013), which were introduced to manage change in dimensional modeling, and how well these fit with current requirements for high performance. Along with the traditional types of dimensions, a simple and new type of dimension is introduced, called a *temporal dimension*. It is based on current research in temporality (Häitgen 2012; Rönnbäck et al. 2010; Čokic, Mahnič, and Kovac 2017), and suffers from none of the refresh anomalies associ-

ated with slowly changing dimensions (R. J. Santos and Bernardino 2008; Silviskas et al. 1998). For most types of slowly changing dimensions the fact table and its dimension tables become temporally dependent, leading to update anomalies and performance degradation (Araque 2003). This is not the case when using temporal dimensions, in which the tables are temporally independent, such that existing relationships are preserved regardless of all changes. The authors, we, recommend that this new type of dimension should become the de-facto standard for managing change in dimensional modeling and, particularly when considering real-time analytics, a growing requirement within data warehousing (Russen, Stadler, and Harper 2014; Arvine, Cui, and Nauck 2005). While much research has been done into strategies for loading data at near real-time (Jing and Deschêch 2009; Vassiliadis and Simitsis 2009), little has been done with respect to finding the most suitable modeling techniques. The temporal dimension and many of the

^{*}Thanks to Up to Change (www.uptochange.com) sponsoring research on ANCHOR and TRANSITIONAL MODELING.

¹ Matteo Golleri and Stefano Rizzi. “A survey on temporal data warehousing”. In: *International Journal of Data Warehousing and Mining (IJDM)* 5.1 (2009), pp. 1–17

² B. Liu. *Uncertainty Theory: An Introduction to Its Axiomatic Foundations*. 2004. Springer-Verlag, Berlin, 2004.

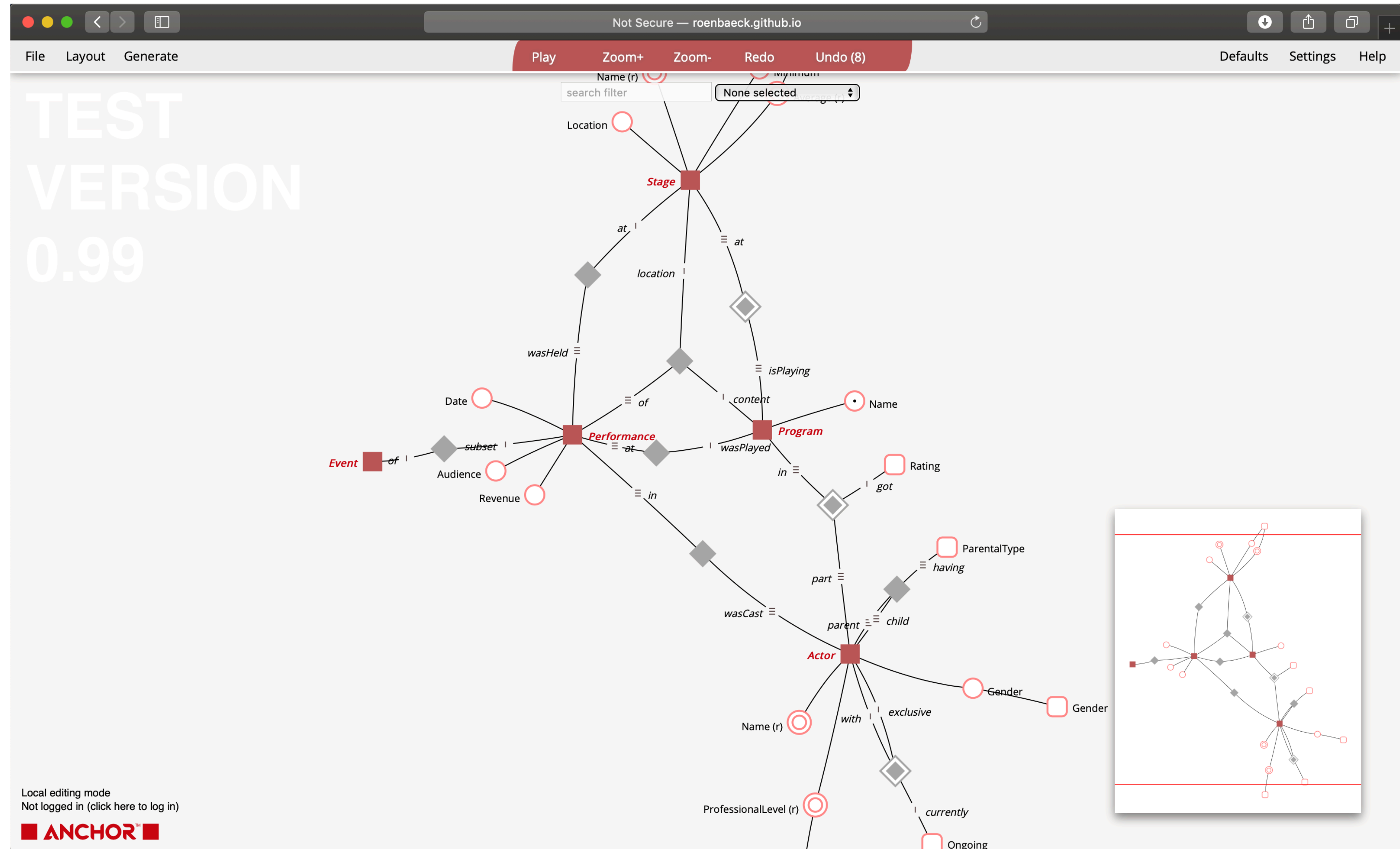
³ Craig W. Fisher and Bruce R. Kingma. “Criticality of data quality as exemplified in two disasters”. In: *Information & Management* 39.2 (2001), pp. 109–116.

⁴ Krishna Kulkarni and Jan-Elke Michels. “Temporal features in SQL: 2011”. In: *ACM Sigmod Record* 41.3 (2011), pp. 34–43.

⁵ Tom Johnston. *Bitemporal data: theory and practice*. Newnes, 2014.

⁶ With gratitude to the consultant company Up to Change (www.uptochange.com) sponsoring research on ANCHOR and TRANSITIONAL MODELING.

www.anchormodeling.com/modeler/latest



github.com/Roenbaeck/sisula

github.com

README.md

sisula

sisula, short for "simple substitution language", is a language for producing text output from XML input.

The current version is built in [JavaScript](#) and should run using [HTA](#) in any Windows version from the last decade. There are no special requirements or dependencies. A legacy version using [JScript](#) in [Windows Scripting Host](#) is also available.

ETL

The ETL branch contains an SQL driven ELT framework for data warehouse automation. This framework can be used with SQL Server and is particularly useful for [Anchor Modeling](#). There is a playlist of video tutorials on how to use it available here: <https://www.youtube.com/playlist?list=PLG6-3kKEOyYIWEaEFzhcARTjqHU6zn1cH>

Sisulator


The sisulator takes an XML file as input and converts this into a JSON-compatible object according to a mapping ruleset. It will then process a number of sisulets as specified in the given directive, which recieve the object as input. The sisulets are parsed and the sisula language substituted to JavaScript/JScript using regular expressions, after which the JavaScript/JScript is evaluated and the output stored.

History

sisula was introduced in [Anchor Modeling](#) in order to replace XSLT for producing text output, and a first JavaScript version of the Sisulator is built into its [modeling tool](#). This version is derived from that work.

© 2019 GitHub, Inc.

[Terms](#) [Privacy](#) [Security](#) [Status](#) [Help](#)



[Contact GitHub](#) [Pricing](#) [API](#) [Training](#) [Blog](#) [About](#)

