Breathing Life into Models: The Next Generation of Enterprise Modeling

Peter Fettke
Saarland University and German Research Center for Artificial Intelligence (DFKI), Germany

Keynote, ICSOFT 2022
Lisbon, Portugal, July 12, 2022
Breathing Life into Models: The Next Generation of Enterprise Modeling

Peter Fettke
Saarland University and
German Research Center for Artificial Intelligence (DFKI), Germany

Wolfgang Reisig
Humboldt-Universität zu Berlin, Germany

Keynote, ICSOFT 2022
Lisbon, Portugal, July 12, 2022
Agenda

1. Preliminary Remarks
2. Future of Enterprise Modeling
3. Conclusions
Enterprise modeling in 1993: Hasso Plattner and Klaus Besier pose with the SAP ERP Reference Model (as a marketing instrument)
Enterprise modeling in 1993: Hasso Plattner and Klaus Besier pose with the SAP ERP Reference Model (as a marketing instrument)

Historical sidenote
- 1990 first Honorary Doctorship from Saarland University
- 1994 first Honorary Professorship for business informatics from Saarland University
- 1997 Honorary Senator of Saarland University

Picture of the ceremony
President of Saarland University (Günther Hönn) and Plattner in 1997
Enterprise modeling today

source: Moody (2005)
Future of enterprise modeling: width vs. depth

mile wide and inch deep
Future of enterprise modeling: width vs. depth

1 mile wide and inch deep

1 inch wide and mile deep (this keynote)
Agenda

1. Preliminary Remarks
2. Future of Enterprise Modeling
3. Implications and Conclusions
What is the most fundamental concept of enterprise modeling?
Change (“everything flows”)?
But: What is “change”? And how to study “change”?

St. Peter’s Square, Vatican City
Most basic understanding of change

INPUT $x$

FUNCTION $f$: 

OUTPUT $f(x)$
"The calculus was the first achievement of modern mathematics and it is difficult to overestimate its importance. I think it defines more unequivocally than anything else the inception of modern mathematics, and the system of mathematical analysis, which is its logical development, still constitutes the greatest technical advance in exact thinking."

*John von Neumann (1947)*

But: Is the calculus adequate to study change of enterprise systems?
Typical assumptions about modeling the world

continuous change

continuum of objects (uncountable set) in total order*

Typical example
- set of real numbers
- typical total order on real numbers

* Let $M$ be a set and $<$ a binary relation on $M$. $<$ is a partial order, iff $<$ is irreflexive and transitive. $<$ is a total order, iff $<$ is a partial order and for all $a, b$ element of $M$ the following assumption holds: $a < b$ or $b < a$ or $a = b$. 
Typical assumptions about modeling the world

<table>
<thead>
<tr>
<th>continuous change</th>
<th>versus</th>
<th>discrete change</th>
</tr>
</thead>
<tbody>
<tr>
<td>continuum of objects (uncountable set) in total order*</td>
<td></td>
<td>discrete set of objects (countable or even finite) in partial order*</td>
</tr>
</tbody>
</table>

Typical example
- set of real numbers
- typical total order on real numbers

Implications
- a total order is “just” a particular case of a partial order
- total order empirically discovered or intentionally designed

* Let M be a set and < a binary relation on M. < is a partial order, iff < is irreflexive and transitive. < is a total order, iff < is a partial order and for all a, b element of M the following assumption holds: a < b or b < a or a = b.
continuum of time is typically assumed
But what is time in enterprise systems?

"We have to consider that all our judgments in which time plays a role are always judgments about simultaneous events. If, for example, I say: 'That train arrives here at 7 o'clock', this means something like: 'The pointing of the small hand of my watch to 7 and the arrival of the train are simultaneous events.'"

Einstein, 1905, p. 893, original in German, translation with deepL
Maybe, events ("change") in enterprise systems are only partially ordered by a cause-effect-relation.
Maybe, events ("change") in enterprise systems are only partially ordered by a cause-effect-relation.
Maybe, events ("change") in enterprise systems are only partially ordered by a cause-effect-relation.
Statics: data and real-life items

`digital stamp with matrix code`
Statics: data and real-life items

digital stamp with matrix code

structure S

sets
letters = \{a, b, c\}
stamps = \{l, m, n, o\}
matrix codes = \{1, 2, 3\}

functions
f: letters \rightarrow stamps
f(a) = m, f(b) = n, f(c) = o

g: stamps \rightarrow matrix codes
\begin{align*}
g(m) &= 1, 
g(n) &= 2, 
g(o) &= 3
\end{align*}
Statics: data and real-life items

digital stamp with matrix code

signature $\Sigma$

<table>
<thead>
<tr>
<th>set symbols</th>
<th>properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>letters</td>
<td>- $f$ is injective</td>
</tr>
<tr>
<td>stamps</td>
<td>- $g$ is injective</td>
</tr>
<tr>
<td>matrix codes</td>
<td>and partial</td>
</tr>
</tbody>
</table>

function symbols

- $f$: letters $\rightarrow$ stamps
- $g$: stamps $\rightarrow$ matrix codes

variables

- $x$: letters

structure $S$

<table>
<thead>
<tr>
<th>sets</th>
<th>functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>letters $= { a, b, c }$</td>
<td>$f$: letters $\rightarrow$ stamps</td>
</tr>
<tr>
<td>stamps $= { l, m, n, o }$</td>
<td>$f(a) = m$, $f(b) = n$, $f(c) = o$</td>
</tr>
<tr>
<td>matrix codes $= { 1, 2, 3 }$</td>
<td>$g$: stamps $\rightarrow$ matrix codes</td>
</tr>
<tr>
<td></td>
<td>$g(m) = 1$, $g(n) = 2$, $g(o) = 3$</td>
</tr>
</tbody>
</table>
Architecture: composition matters
Architecture: composition matters
Completing the example: stamp with matrix code

**set symbols**
- letters
- stamps
- matrix codes

**properties**
- f is injective
- g is injective and partial

**function symbols**
- f: letters → stamps
- g: stamps → matrix codes

**variables**
- x: letters

**sets**
- letters = \{a, b, c\}
- stamps = \{l, m, n, o\}
- matrix codes = \{1, 2, 3\}

**functions**
- f: letters → stamps
  - f(a) = m, f(b) = n, f(c) = o
- g: stamps → matrix codes
  - g(m) = 1, g(n) = 2, g(o) = 3
Completing the example: stamp with matrix code

**signature**

- **set symbols**
  - letters
  - stamps
  - matrix codes

- **function symbols**
  - set of functions:
    - f: letters → stamps
    - g: stamps → matrix codes

**properties**

- f is injective
- g is injective and partial

**variables**

- x: letters

**interface**

- three modules:
  - sender
  - postal service
  - receiver

**three modules**

- sender • postal service • receiver

**causal order**

**composed module**

**structure**

- **sets**
  - letters = \{a, b, c\}
  - stamps = \{l, m, n, o\}
  - matrix codes = \{1, 2, 3\}

- **functions**
  - f: letters → stamps
    - f(a) = m, f(b) = n, f(c) = o
  - g: stamps → matrix codes
    - g(m) = 1, g(n) = 2, g(o) = 3

**schema**

**instatiation**

**system**

**run**

**the run of letter a**

```
<table>
<thead>
<tr>
<th></th>
<th>outbox</th>
<th>post</th>
<th>postbox</th>
<th>forward</th>
<th>delivery box</th>
<th>deliver</th>
<th>inbox</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>a</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>a</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>a</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```
How to understand the world (summary)?

A comprehensive conceptual model of a computer-integrated system

(1) is structured: architecture composition matters!

(2) includes data and real-life items: statics objects matter!

(3) is locally updated: dynamics causality matters!

Modeling in computer science is about changing the perspective:

• classical theoretical computer science and software engineering start with computers,

• modeling starts with the considered system and the problem.
Restaurant (architecture)

abstract module

three composed modules
Restaurant (statics)

**Signature**

- **Set symbols**
  - $C$: clients
  - $T$: tables
  - $M$: order items (called menu)
  - $D$: meal items

- **Function symbols**
  - $f$: $D \rightarrow M$
  - $g$: $Q \rightarrow P$

- **Properties**
  - For $A \subseteq D$:
    - $g(A) = \{f(a) \mid a \in A\}$

- **Variables**
  - $c$: $C$
  - $t$: $T$
  - $x$: $P$
  - $y$: $D$
  - $y$: $Q$

**Structure**

- **Sets**
  - $C$: all persons with an id card
  - $T$: $\{t_1, t_2, t_3, t_4\}$
  - $M$: $\{\text{rice, meat, salad}\}$
  - $D$: $\{\text{rice, rice, meat, salad}\}$

- **Functions**
  - $f$: $D \rightarrow M$
    - $f(\text{rice}) = \text{rice}$
    - $f(\text{meat}) = \text{meat}$
    - $f(\text{salad}) = \text{salad}$

- **Derived symbols**
  - $P$: $\{\{\text{rice, meat}\}, \{\text{rice, salad}\}\}$
  - $Q$: $\{\{\text{rice, meat}\}, \{\text{rice, salad}\}\}$

- **Functions** (from $P$ to $Q$)
  - $g$: $P \rightarrow Q$
    - $g(\{\text{rice, meat}\}) = \{\text{rice, meat}\}$
    - $g(\{\text{rice, salad}\}) = \{\text{rice, salad}\}$
Restaurant (dynamics)

instantiation, a run
Retailer (architecture)

six composed modules
Retailer (dynamics)

One instantiation of the customer module

A. Ordering customer: Ute

B. Ute submits purchase order: (Ute, (p1, p2, p2))

C. P. order copy: (Ute, (p1, p2, p2))

D. Ute receives message: (Ute, (p1, p2, p2))

E. Expected delivery: (Ute, (p1, p2, p2))

F. Ute accepts overall delivery: (Ute, (shoes, shoes, hat, hat))

G. Ute opens partial delivery: (Ute, (hat))

H. Del. goods: (Ute, shoes)

I. Del. goods: (Ute, shoes)

J. Del. goods: (Ute, hat)

K. Del. goods: (Ute, hat)

L. Accepted overall delivery: (Ute, (shoes, shoes, hat, hat))

Items:

- p1 = "shoes", 2
- p2 = "hat", 1

Article list:

- {p1, p2, p2}
Retailer (dynamics)

instantiation, a run
Retailer (statics)

ground sorts
KN customers
AR articles
WA goods
TE dates
SP freight forwarders

function symbols
f : WA → AR
f : WM → AM
(‘) : AP → AM
(‘) : AL → AM

derived sorts
AP = AR × N items
AL = M(AP) article lists
AM = M(AR) sets of articles
WM = M(WA) sets of goods

constant symbols
p1, p2: AP ordered article positions
B: P(KN) ordering customers
G: AL initially listed articles
H: WM initially available goods
R: P(SP) available freight forwarders

variables
k: KN
x: AR
X, Y: AL
Z: WM
t: TE
w: WA
s: SP
m, n, p: N

properties
(a, n)’ = n[a] for (a, n) ∈ AP
[p1, ..., pn]’ = p1’ + ... + pn’ for [p1, ..., pn] ∈ AL

structure

ground sorts
KN = {Ute, Max} customers
AR = {"shoes", "hat", "pants"} articles
WA = {shoes, hat, pants} goods
TE = {12/23, 12/24} dates
SP = {Maier, Müller, Schulz} freight forwarders

signature

function symbols
f : WA → AR
f : WM → AM
(‘) : AP → AM
(‘) : AL → AM

variables
k: KN
x: AR
X, Y: AL
Z: WM
t: TE
w: WA
s: SP
m, n, p: N

properties
(a, n)’ = n[a] for (a, n) ∈ AP
[p1, ..., pn]’ = p1’ + ... + pn’ for [p1, ..., pn] ∈ AL

schema

G’ = f(H)
Agenda

1. Preliminary Remarks
2. Future of Enterprise Modeling
3. Implications and Conclusions
Two faces of computer science and Dijskra’s firewall

application face

firewall

technological face

pleasentness problem

correctness problem
Two faces of computer science and next-generation modeling

- **Application face**
  - Pleasantness problem
  - "Bridge":
    - Freely-selectable abstraction level
    - Relevant real- and imagined world (architecture, statics, dynamics)
  - Model

- **Technological face**
  - Correctness problem
Summing up: What is next generation enterprise modeling all about?

**classical computer science**

modules and composition: merge “equal” interface elements

statics (data, items): symbolic representation

dynamics: steps

... yes, but ...  
yes, however not one interface

... adjusted  
but two!

... such as

composition calculus

... technically

Don’t hesitate!

Join the research program and development project

classical computer science

• jump in the right direction
• but fall short

architecture  
statics  

THE WORLD

THE MODEL
Thanks for your attention! Questions?

Contact
Professor Dr. Peter Fettke
Universität des Saarlandes und
Deutsches Forschungszentrum für Künstliche Intelligenz (DFKI)
Campus D 3 2
66123 Saarbrücken, Germany
peter.fettke@dfki.de
Telefon: +49 681 85775-5142, Fax: -3696