

Business Processes Modelling

MPB (6 cfu, 295AA)

Roberto Bruni

<http://www.di.unipi.it/~bruni>

01 - Introduction





English vs Italian



Accedi

Traduttore

Disattiva traduzione istantanea



Inglese Italiano Francese Rileva lingua ▼



Italiano Inglese Spagnolo ▼

Traduci

Methods for the specification and verification of business processes



Metodi per la specifica e verifica dei processi aziendali



Suggerisci una modifica

Classes

Every

Monday: 11:00-13:00, room A1

Wednesday: 09:00-11:00, room A1

Who am I?

<http://www.di.unipi.it/~bruni>

bruni@di.unipi.it

Office hours
Wednesday 14:00-16:00
(or by appointment)

Who are you?

First Name:	John
Last Name:	Smith
Enrollment number	123456
email:	john.smith@email.com
Bachelor degree:	Comp. Sci., Stanford, US
MSc course of enrollment:	Data Science & BI
Subjects of interest:	Statistics

Please, send your data to bruni@di.unipi.it
with object "MPB"

What is a BP?

Any set of steps aimed to reach some outcome

from opening an account



to processing a custom order

What is BPM about?



Course objectives

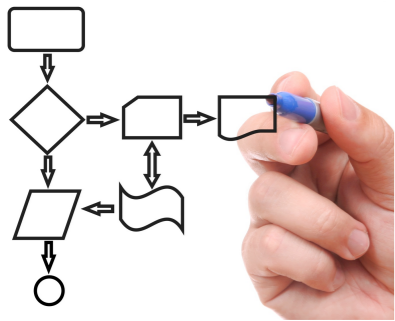
Key issues in Business Process Management (patterns, architectures, methodologies,...)



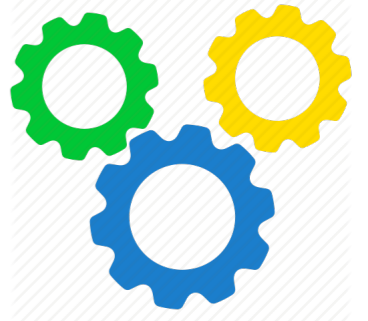
Analysis techniques and correctness by construction (soundness, boundedness, liveness, free-choice,...)



Graphical languages & visual notation (BPMN, EPC, BPEL, ...)



Tool-supported verification (WoPeD, ProM, YAWL, ...)



Structural properties, behavioural properties and problematic issues (dead tasks, deadlocks, ...)



Performance analysis (bottlenecks, simulation, capacity planning,...)



Formal models (automata, Petri nets, workflow nets, YAWL, ...)



Process mining (discovery, conformance checking, enhancement,...)

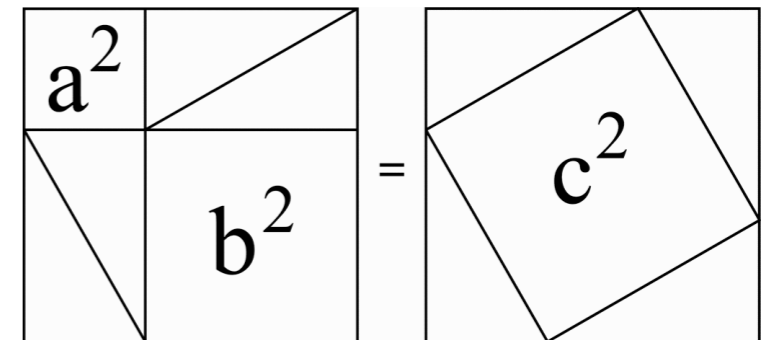


Course activities



attend classrooms:
ask questions!
(sleep quietly)

learn theorems:
(drink many coffees)



do some thinking:
solve ALL exercises

deliver a project:
practice with concepts,
experiment with tools



give the exam:
time for a party!

Main Textbook



Mathias Weske

Business Process Management: Concepts, Languages, Architectures
(2nd ed.) Springer 2012

<http://bpm-book.com>

Mathias Weske

Business
Process
Management

Concepts, Languages, Architectures

Second Edition

 Springer

Other Textbooks

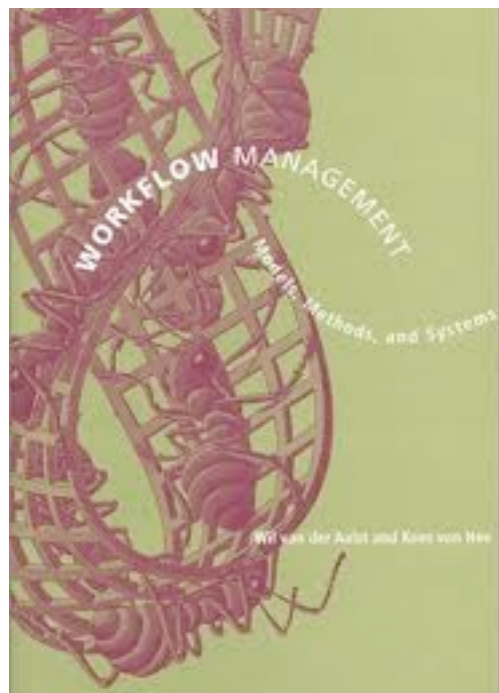


Joerg Desel and Javier Esparza

Free Choice Petri Nets

Cambridge Tracts in Theoretical Computer Science 40, 1995

<https://www7.in.tum.de/~esparza/bookfc.html>



Wil van der Aalst, Kees van Hee

Workflow Management: Models, Methods, and Systems

MIT Press (paperback) 2004

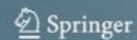
<http://www.workflowcourse.com>

Other Textbooks



Fundamentals of
**Business Process
Management**

Marlon Dumas
Marcello La Rosa
Jan Mendling
Hajo A. Reijers



Marlon Dumas, Marcello La Rosa, Jan Mendling, Hajo Reijers
Fundamentals of Business Process Management
Springer 2013

<http://fundamentals-of-bpm.org>



Wil M. P. van der Aalst

**Process
Mining**

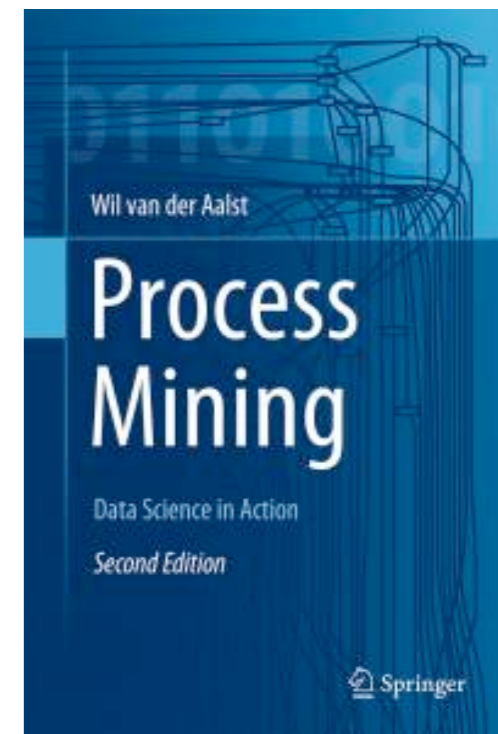
Discovery, Conformance and
Enhancement of Business Processes



Wil van der Aalst
Process Mining
Springer 2011 / 2016

<http://springer.com/978-3-642-19344-6>

<http://springer.com/978-3-662-49850-7>



Wil van der Aalst

**Process
Mining**

Data Science in Action

Second Edition



Main resources

- Petri nets

- <http://www.pnml.org>

- <http://www.informatik.uni-hamburg.de/TGI/PetriNets>

- BPMN

- <http://www.bpmn.org>

- BPEL

- <http://www.oasis-open.org/committees/wsbpel>

- Workflow Patterns

- <http://www.workflowpatterns.com>

Main resources (tools)

- Woped

- <http://www.woped.org>

- ProM

- <http://http://www.promtools.org/prom6>

- <http://www.win.tue.nl/woflan>

- Diagnosing workflow processes using Woflan. H.M.W.Verbeek, T.Basten, W.M.P. van derAalst. Computer J. 44(4): 246-279 (2001)

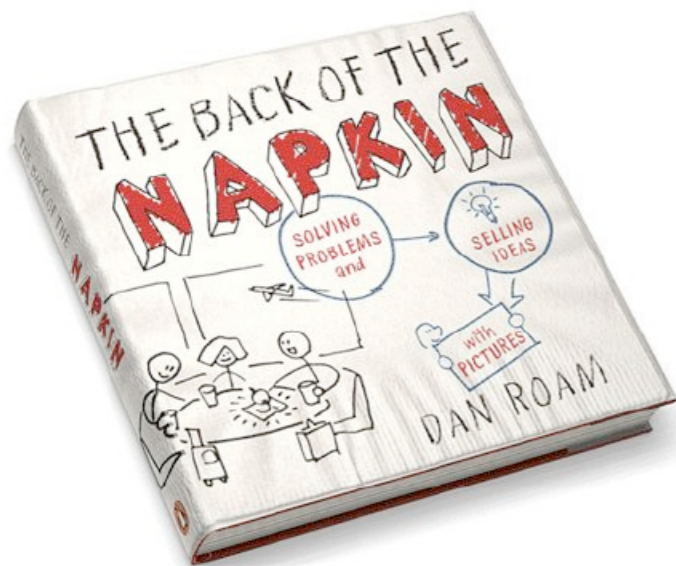
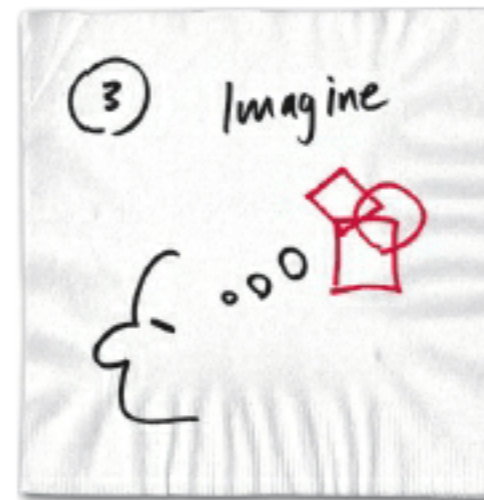
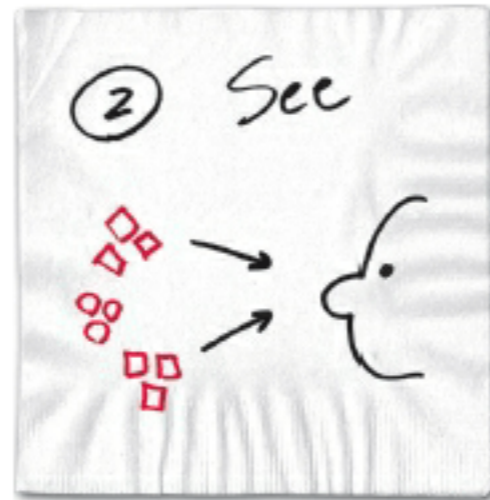
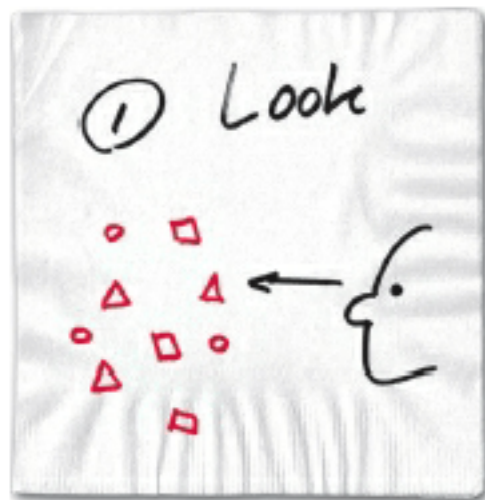
- <http://wwwis.win.tue.nl/~wvdaalst/publications/p135.pdf>

- YAWL

- <http://www.yawlfoundation.org>

What is BPM about?

Giving shape to ideas, organizations, processes, collaborations, practices



To analyse them

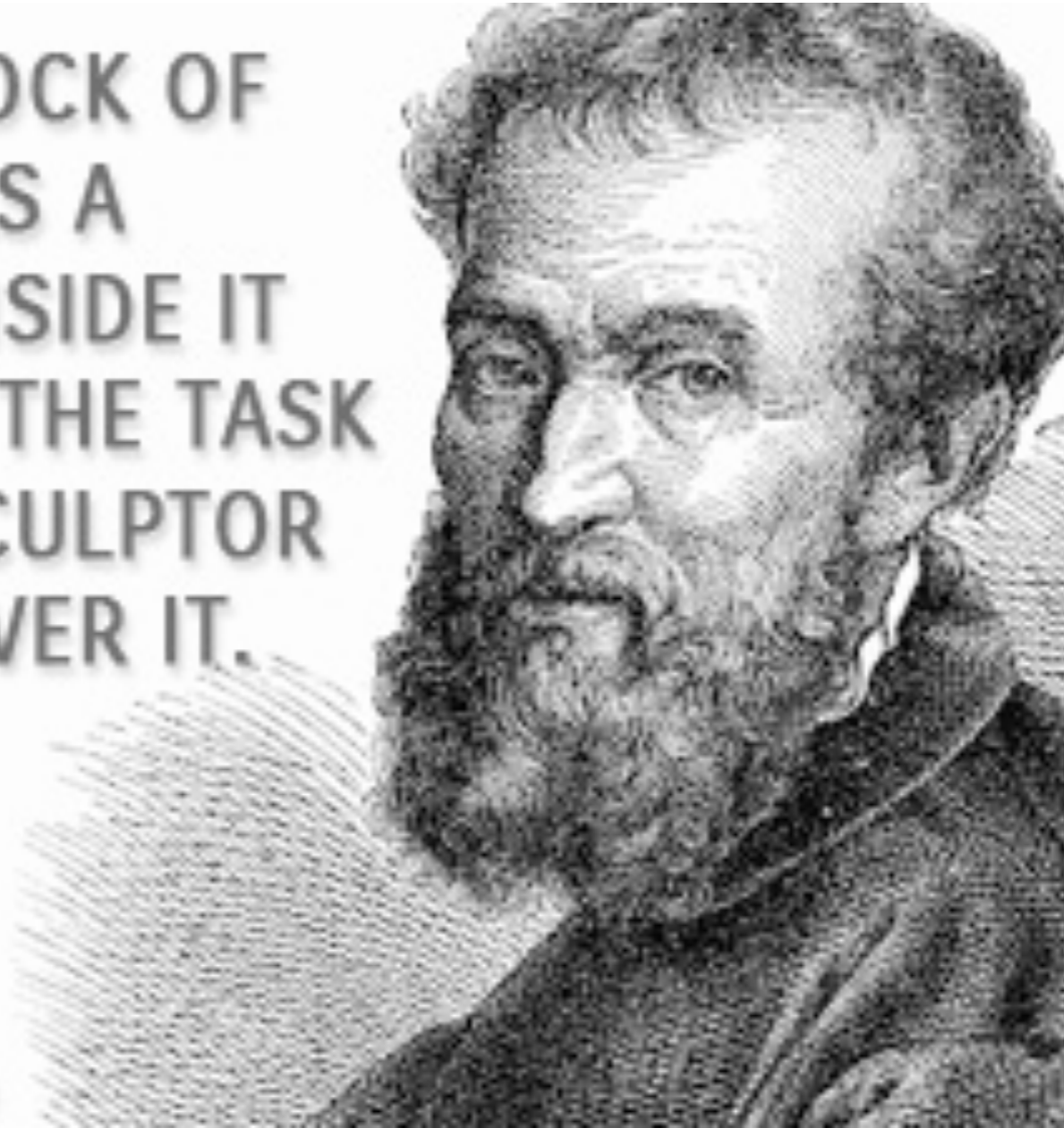
To communicate them to others

To change them if needed

Quoting Michelangelo

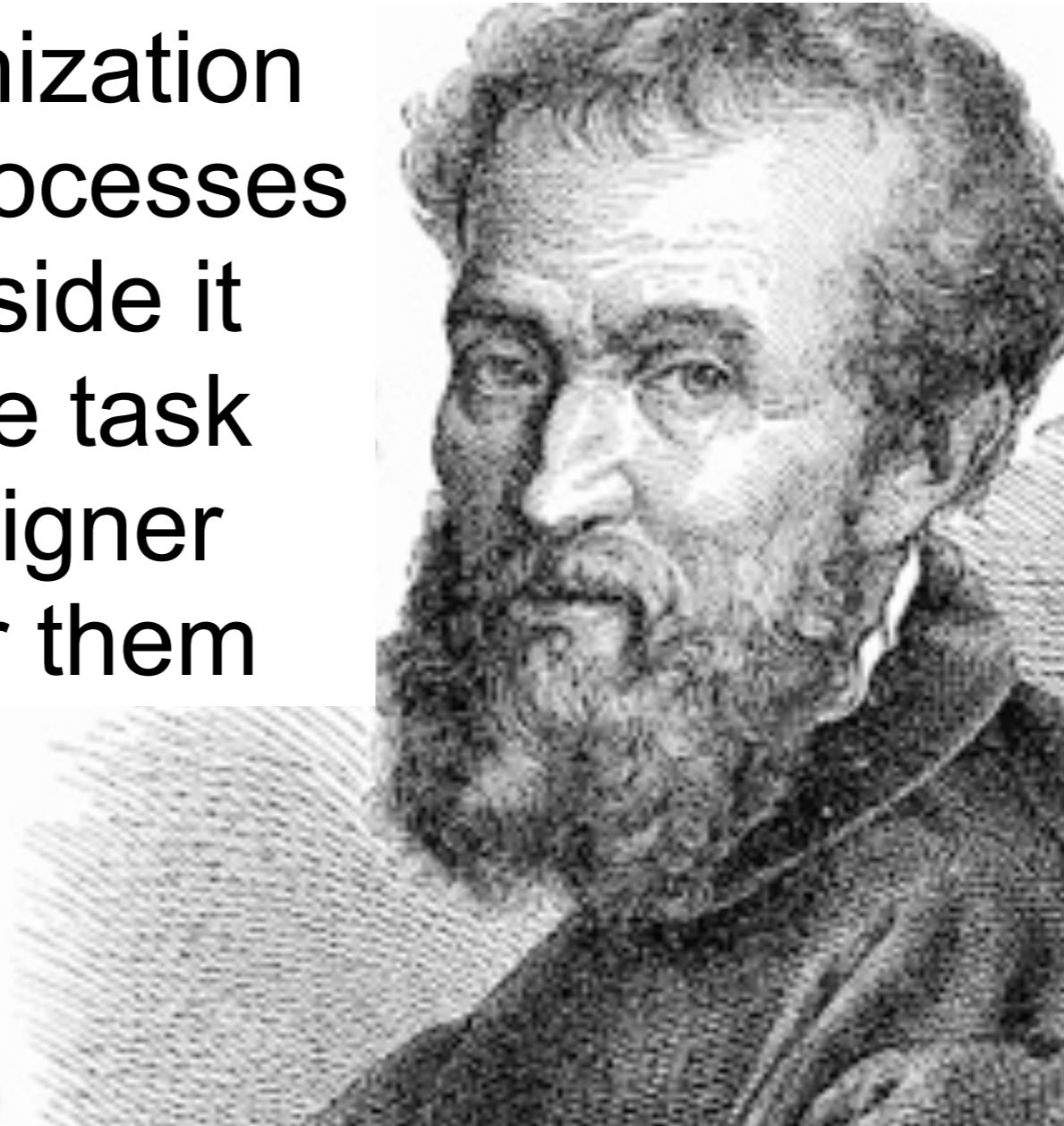
EVERY BLOCK OF
STONE HAS A
STATUE INSIDE IT
AND IT IS THE TASK
OF THE SCULPTOR
TO DISCOVER IT.

Michelangelo



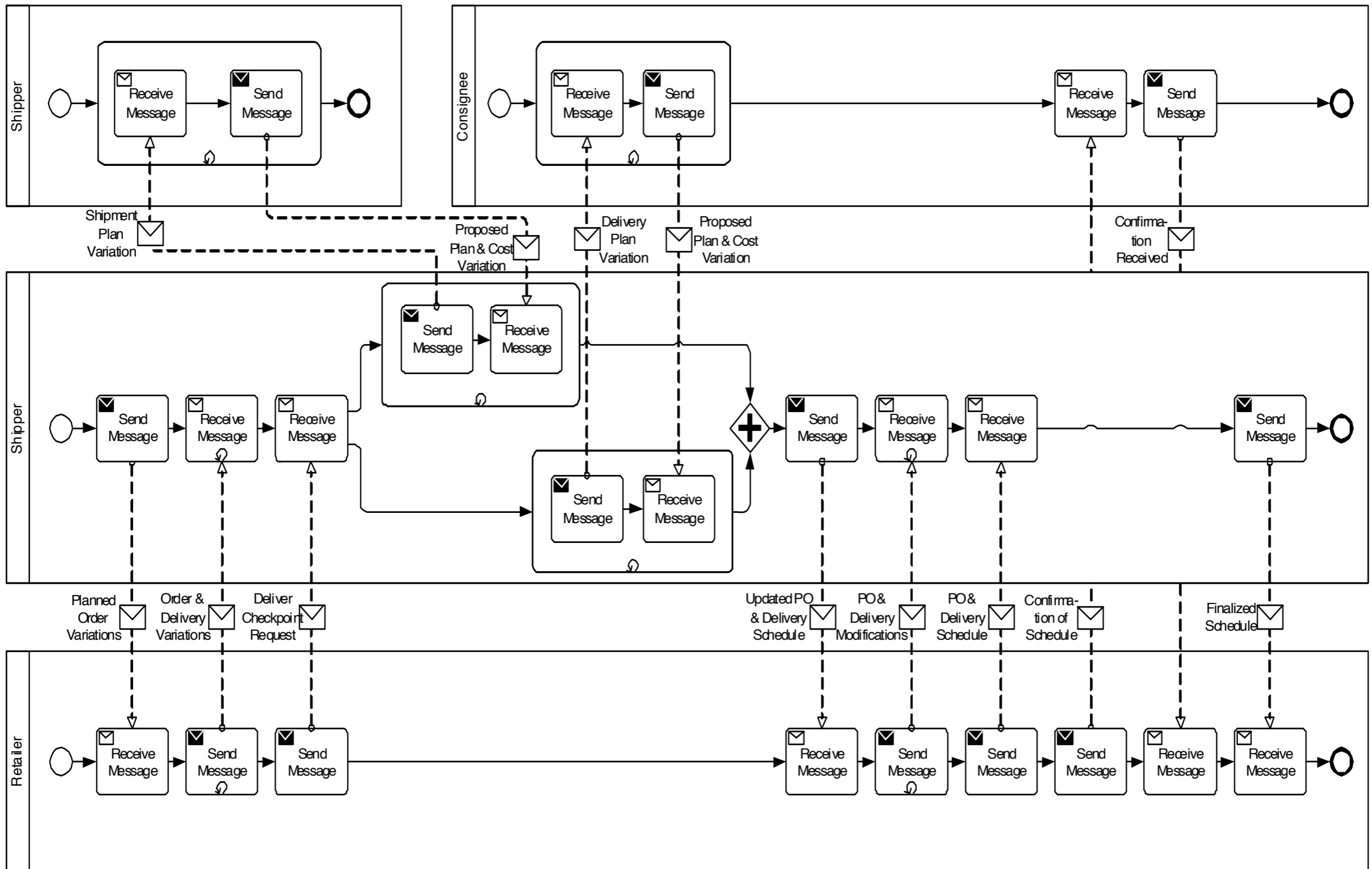
Quoting Michelangelo

Every organization has some processes running inside it and it is the task of the designer to discover them



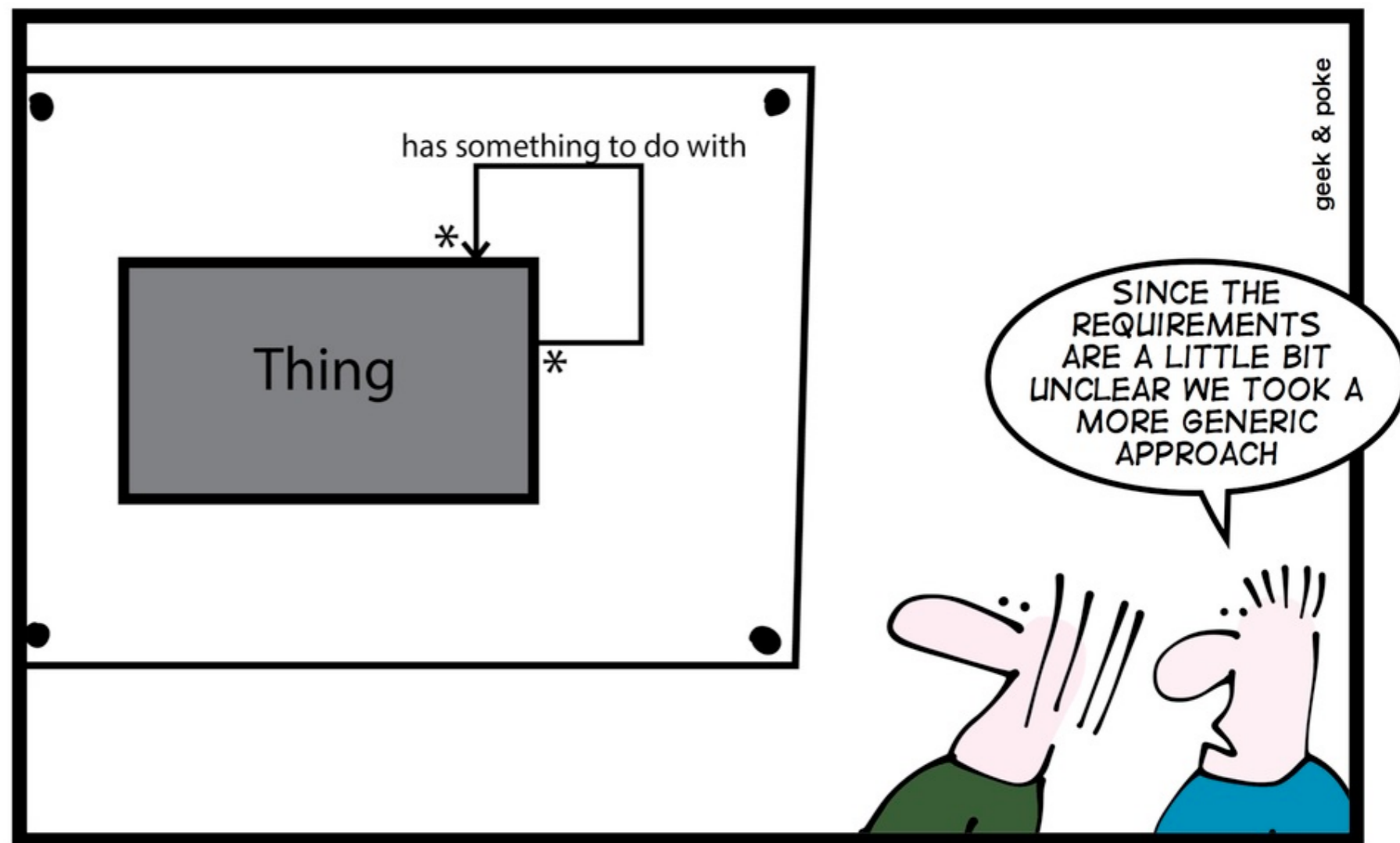
Michelangelo

A taste of BPMN



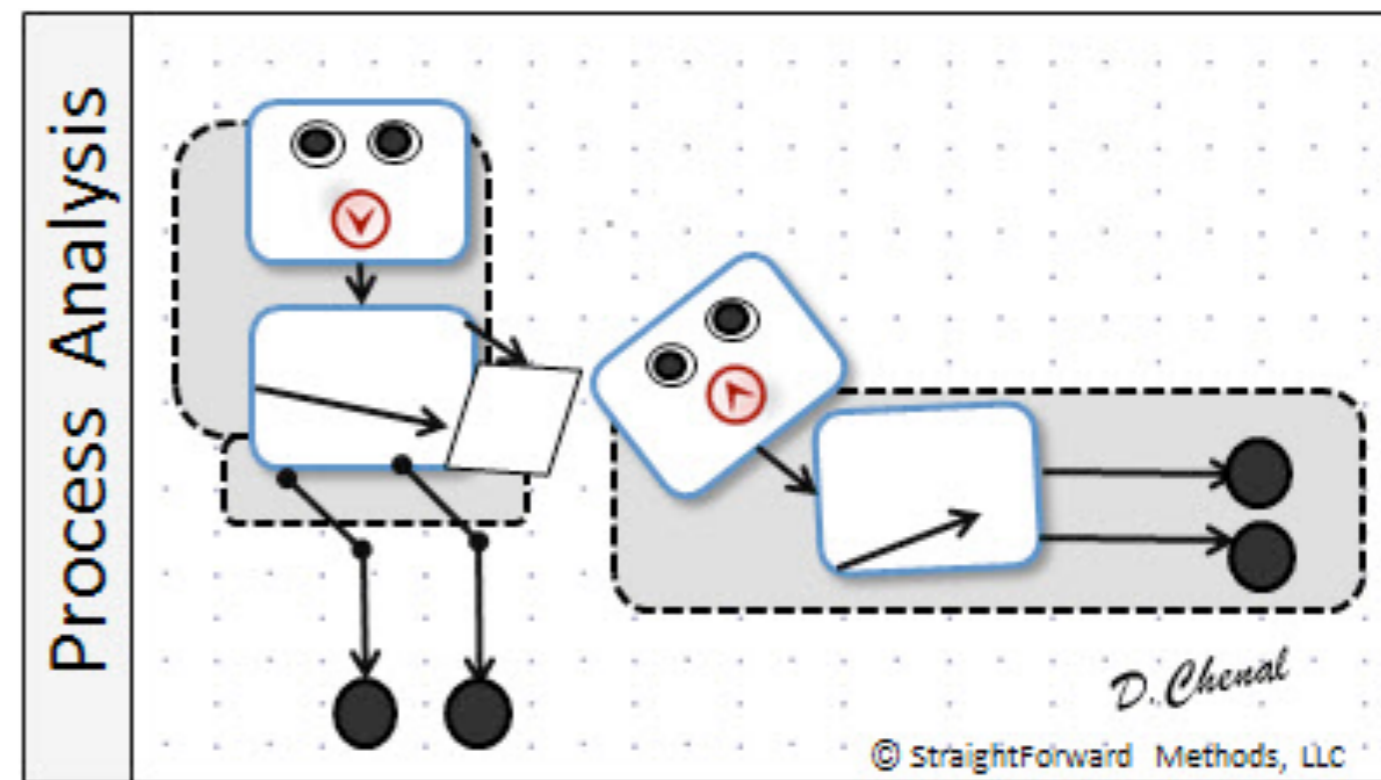
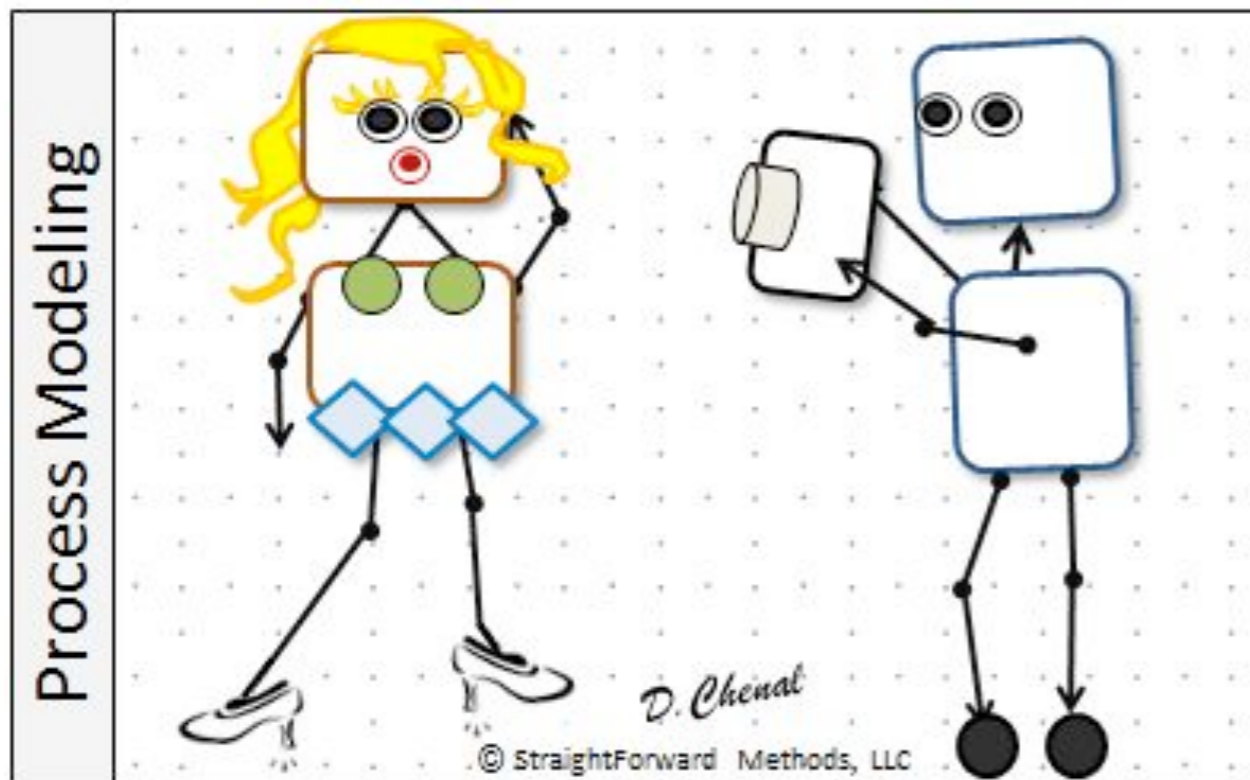
Data and processes

Traditionally, information systems used **information modelling** as a starting point



Data and processes

Nowadays, **processes** are of equal importance and need to be supported in a systematic manner

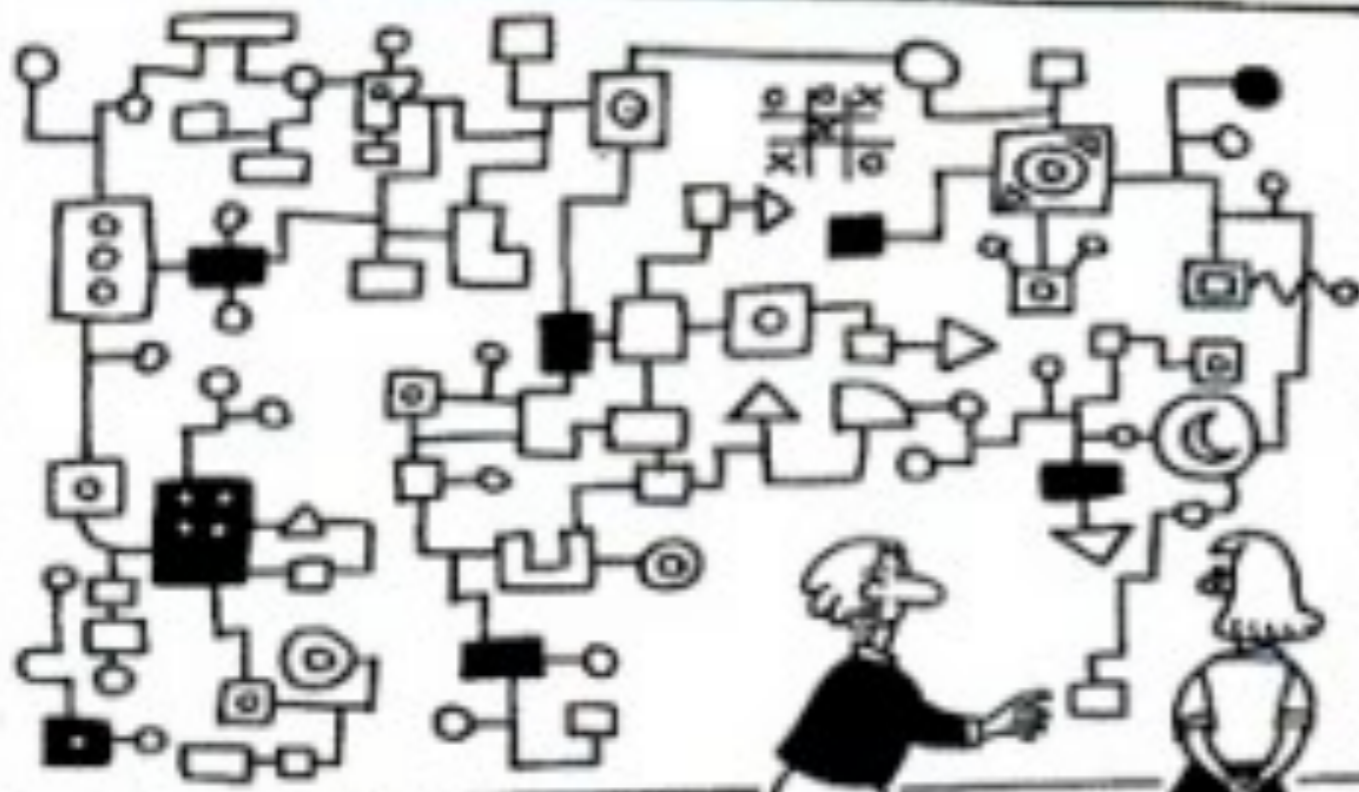


Motivation

- Each product is the outcome of a number of activities performed
- Because of modern communication facilities:
 - traditional product cycles not suitable for today's dynamic market
- Competitive advantages of successful companies:
 - the ability to bring new products to the market rapidly and
 - the ability to adapt an existing product at low cost
- Business processes are the key instrument:
 - to organize these activities
 - to improve the understanding of their relationships
- IT is an essential support for this aim

Workflow wave

WORKFLOW REDESIGN



Geff

In the mid-nineties, workflow management systems aimed to the automation of structured processes

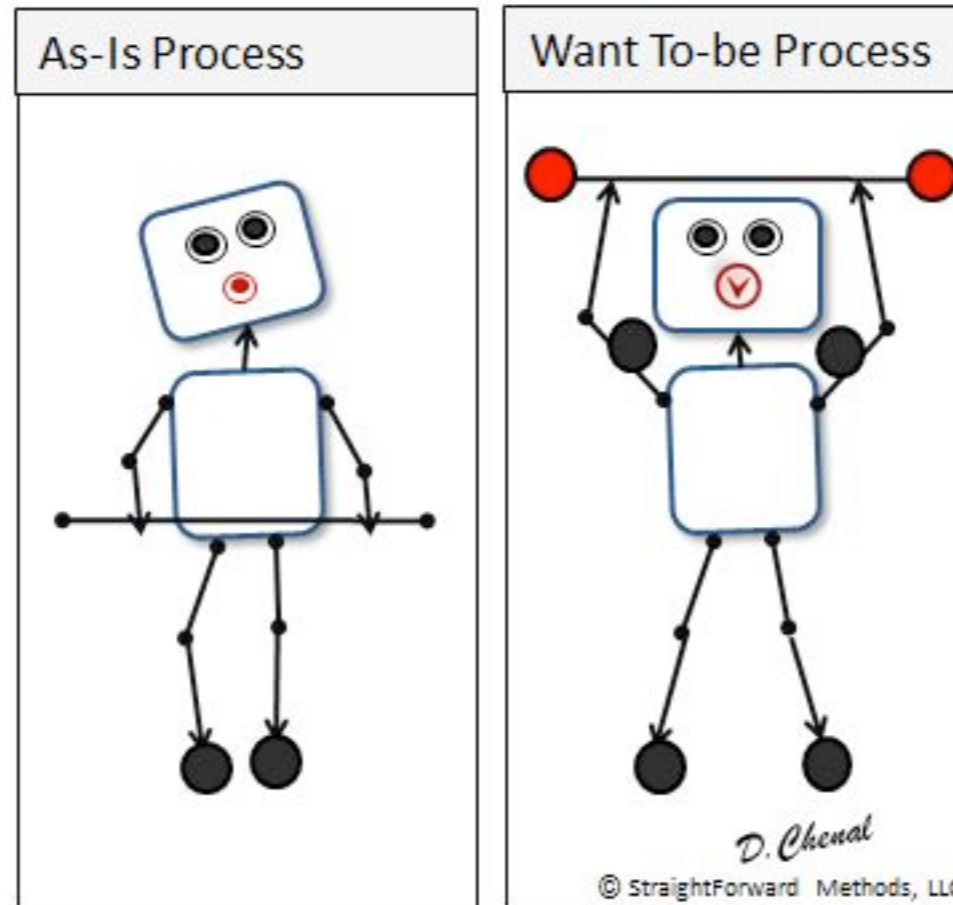
but their application was restricted to only a few application domains

"And this is where our ED workflow redesign team went insane."

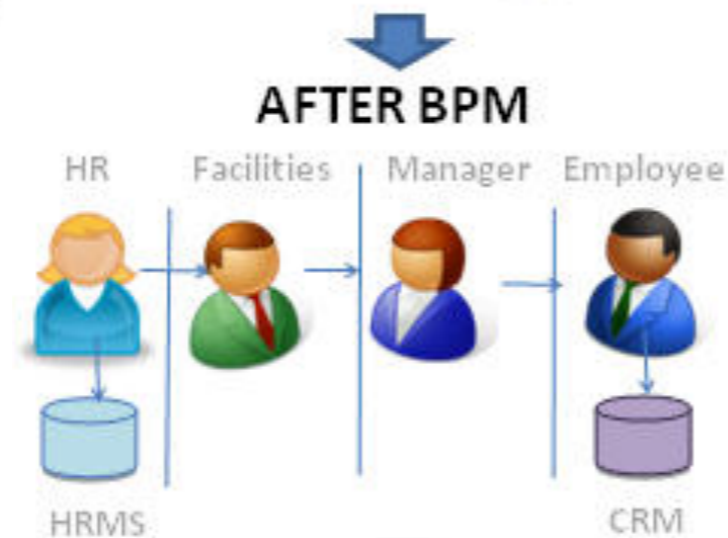
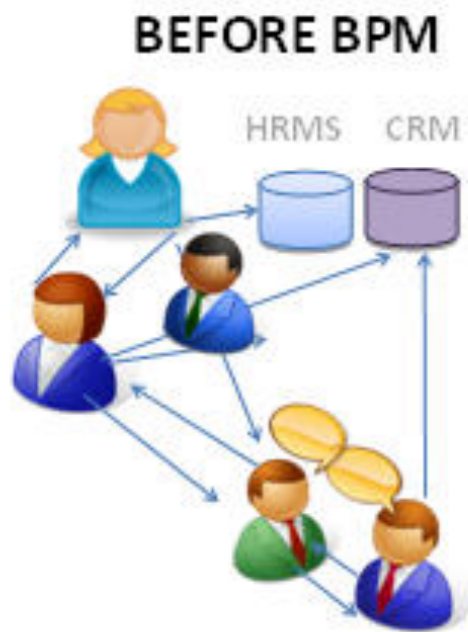
Process awareness

BPM moves
from
workflow management systems
(intra-organization)

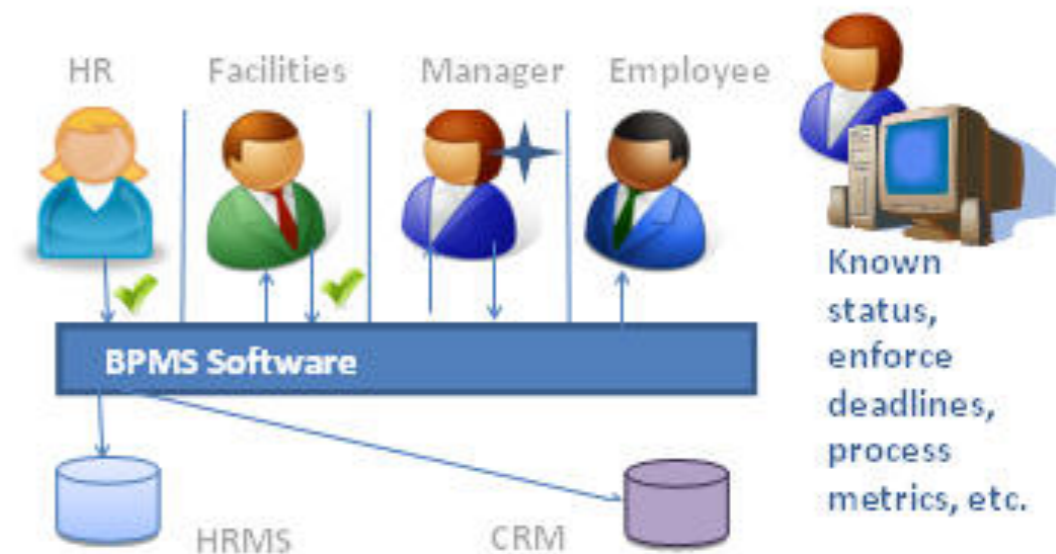
to the broader perspective of
process-aware information
systems
(inter-organizations)



Process awareness



AFTER BPMS



Know end goal
Handled
differently every
time

Benefits of BPM

automate workflow and
orchestrate processes,
reduce risk of errors,
provide metrics,
real time status,
enforce deadlines,
validate data,
reduce training costs,

...

What is the BPM maturity of your organization?

1 initial

No structured BPM activities in the area of responsibility of the stakeholder.

2 awareness

Awareness of BPM exists in the organization.

(Planning) activities have started for the definition of the subject.

3 defined

BPM is defined.

Implementation is yet missing or ongoing.

4 managed

BPM is implemented. (People assigned. Communication to relevant people done. Training done, etc.)

5 excellence

BPM is implemented enterprise-wide. A continuous review & improvement process is implemented to exchange lessons-learned & address required changes proactively.



© 2008 IDS SCHEER

Why BPM?

Highly relevant for
practitioners

Offers many challenges for
software developers and
computer scientists

BPM angles

Analysis: simulation,
verification, process mining, ...

Influences: business aspects,
social aspects, training, education, ...

Technologies: interoperability,
standardization efforts,
service orientation, ...

Essential concepts

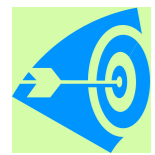
Different educational
backgrounds and interests
are in place

This course is not about a
particular XML syntax (e.g., BPMN)
or tool (e.g. ProM)

It is about using some process
languages to describe, single out,
relate, compare essential concepts

Which target?

Formal methods people



- investigate properties
- detect and correct deficiencies
- abstract from "real world"

Software develop people



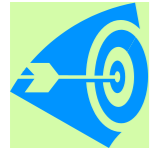
- provide robust and scalable sw
- integration of existing sw
- look at new technology trends

Business admin people

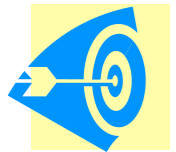


- increase customer satisfaction
- reducing costs
- establishing new products

Aim



Robust and correct realization of



business processes in software that






increases customer satisfaction and

ultimately contributes to the competitive

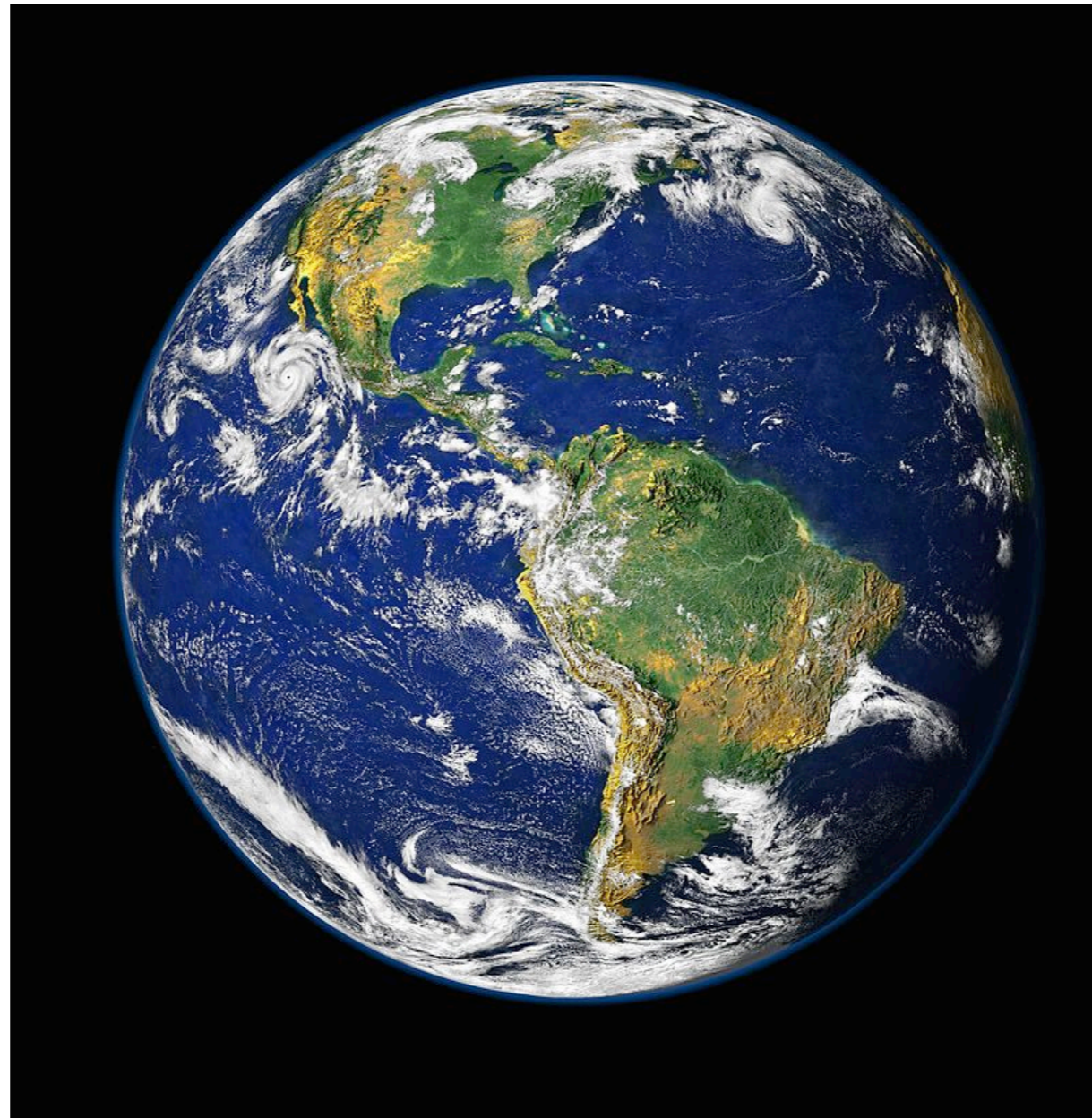
advantage of an enterprise

Abstraction

- Business admin people
 -  – IT as a subordinate aspect (for expert technicians)
 - This course: too much math!
- Software develop people
 -  – Current technology trend as main concern
 - This course: too abstract!
- Formal methods people
 -  – Underestimate business goals and regulations
 - This course: too much handwaving!

Abstraction as the key to achieve some common understanding, to build a bridge between views...

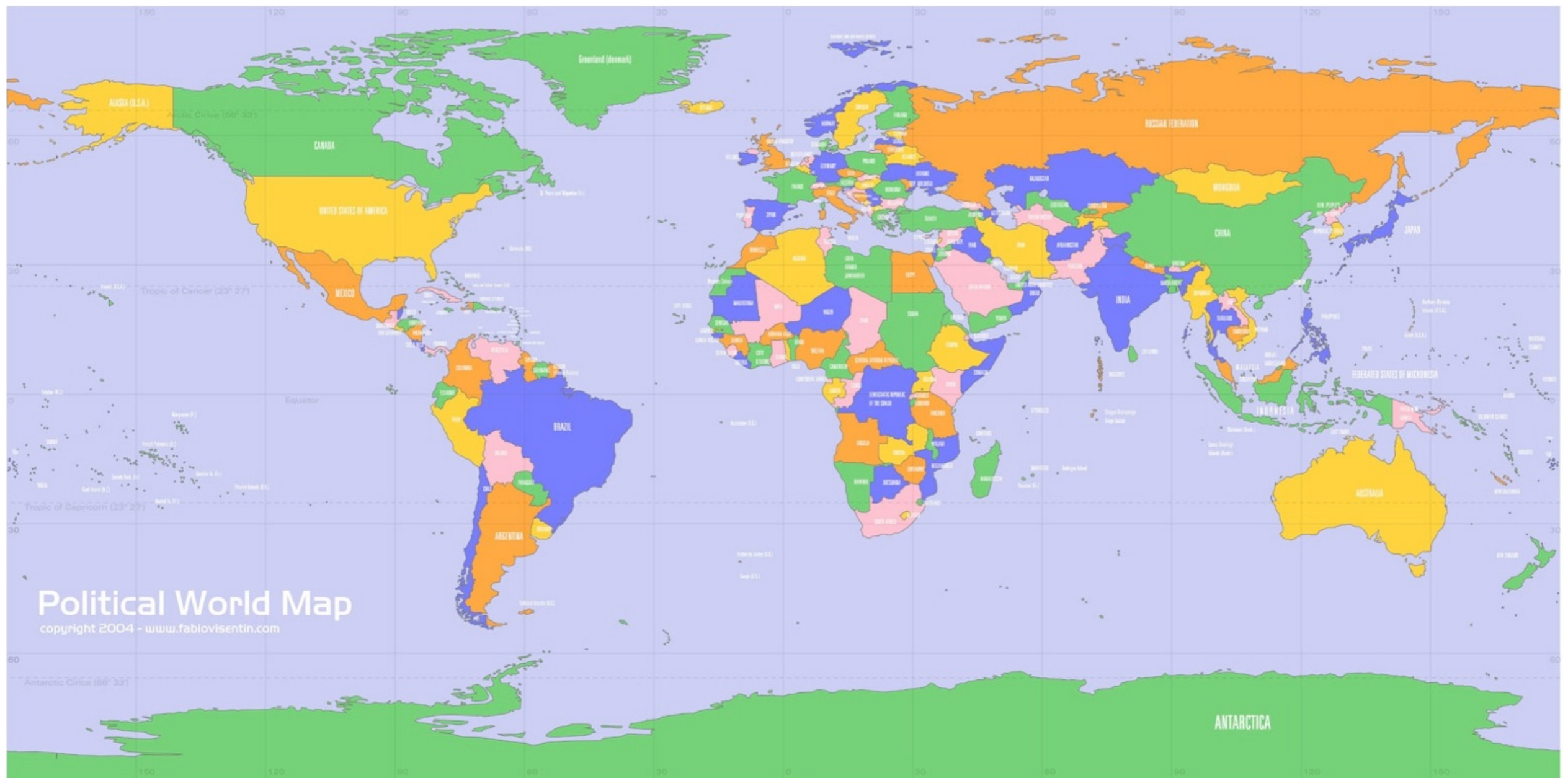
Levels of abstractions



Levels of abstractions



Levels of abstractions



Levels of abstractions



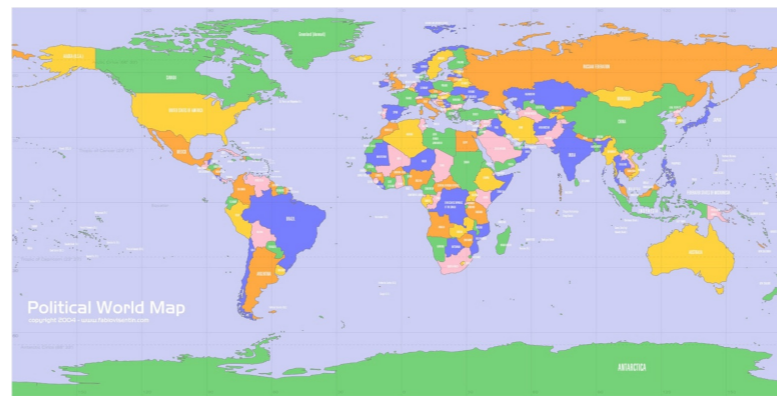
Levels of abstractions



Levels of abstractions

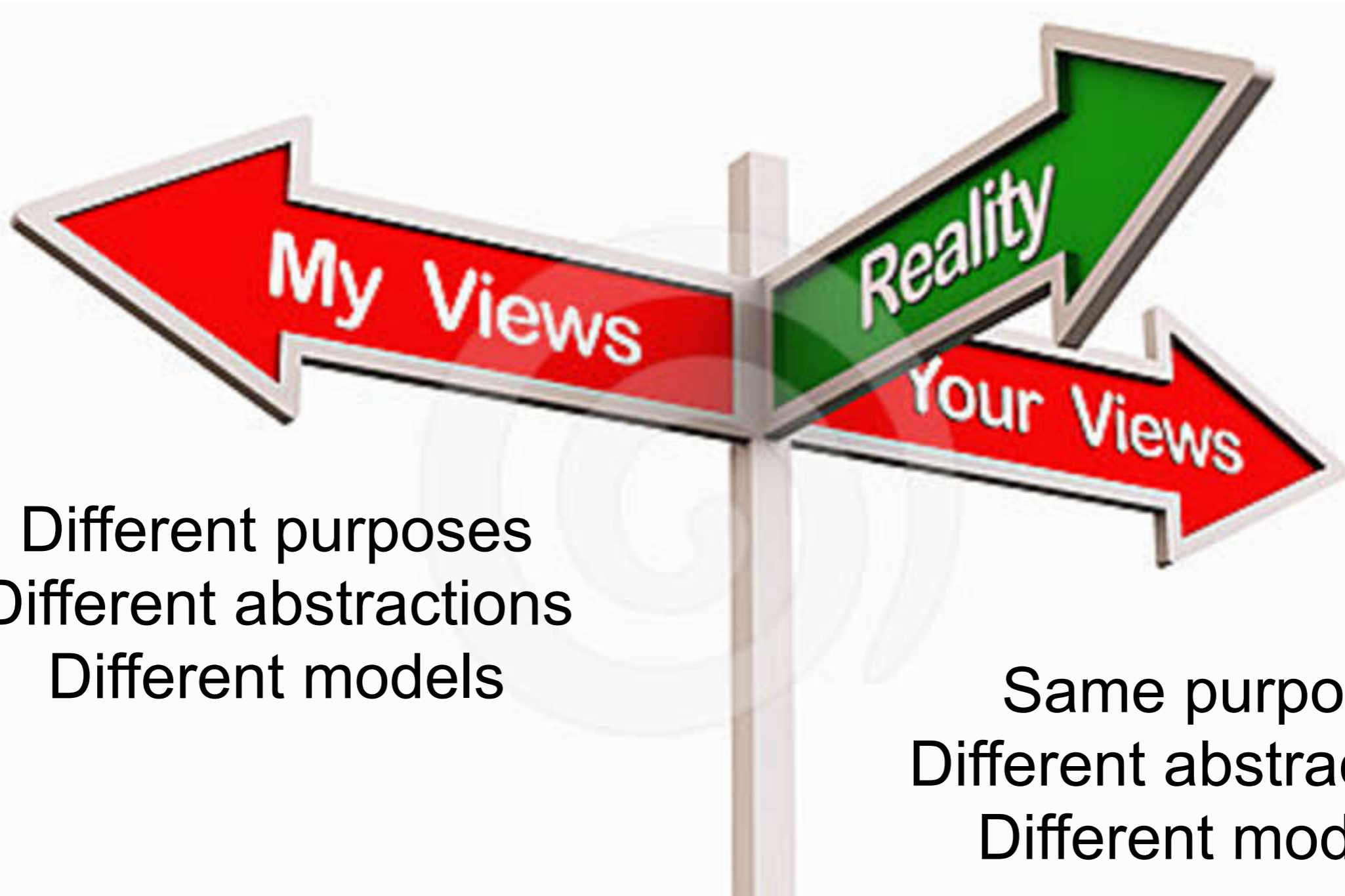


Levels of abstractions



One object, many views

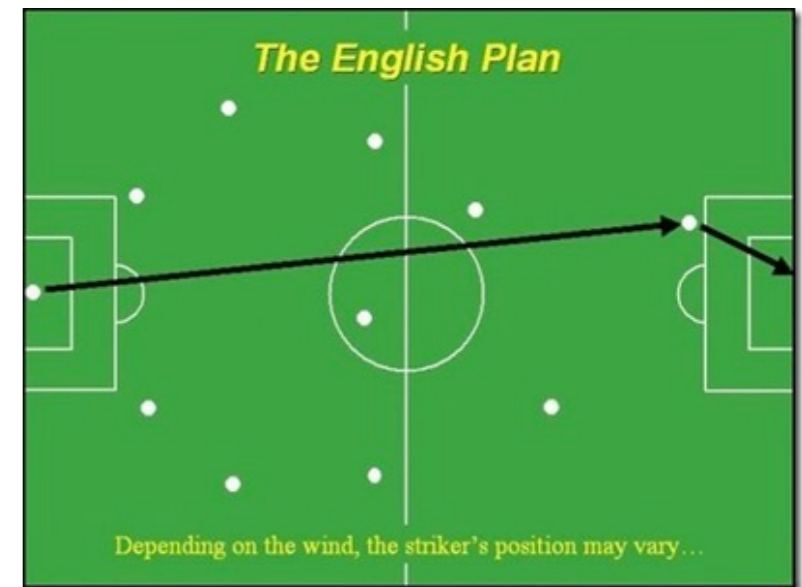
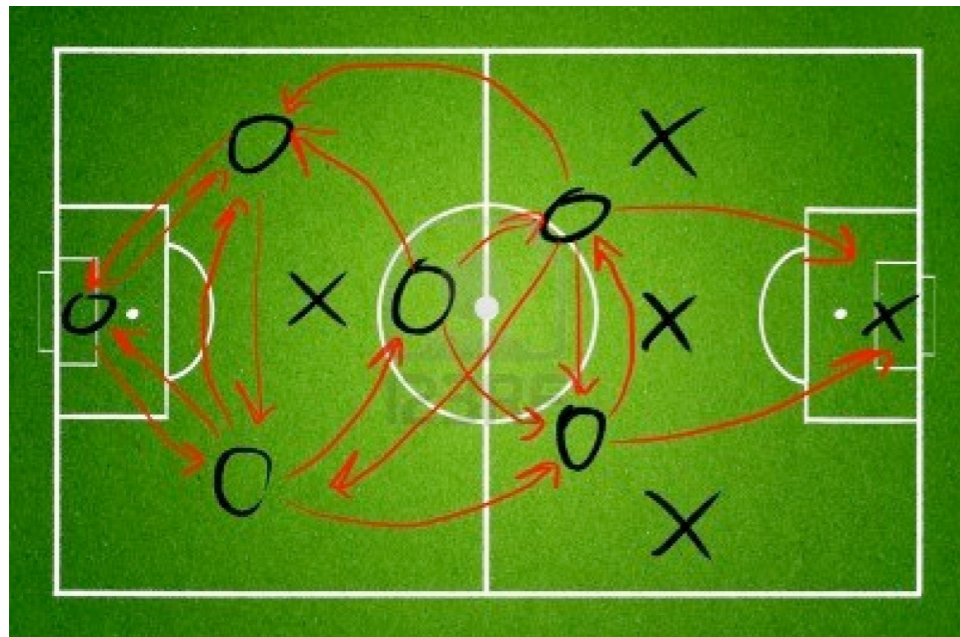
Different views are common



Different purposes
Different abstractions
Different models

Same purpose
Different abstractions
Different models

Everybody wants to be the Italian soccer team coach



What about the adversaries?

Can we find out their plan?

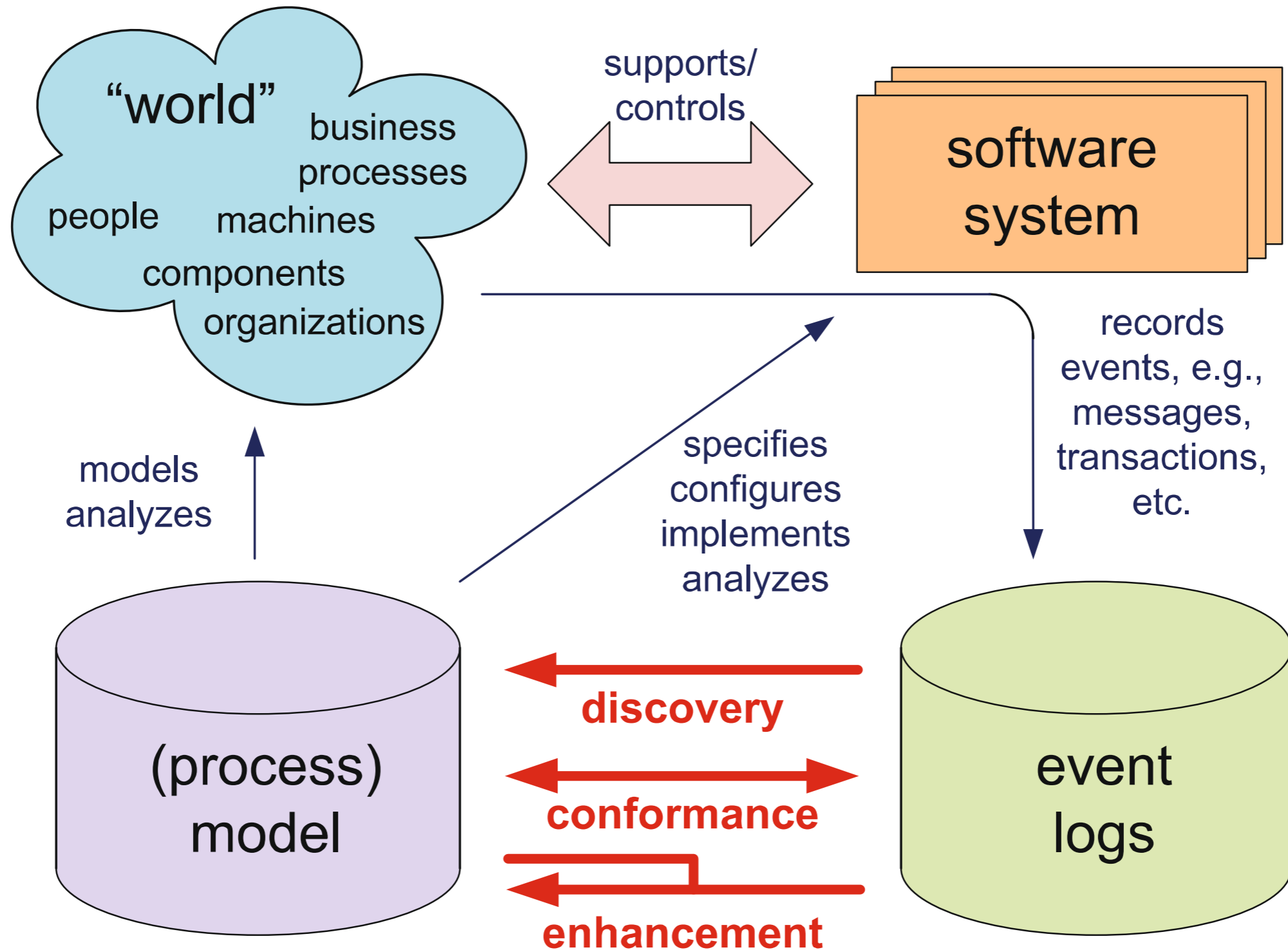


Knowing it would
be quite helpful

Any idea how to?

(abstractions can be designed but can also be derived)

A taste of Process Mining

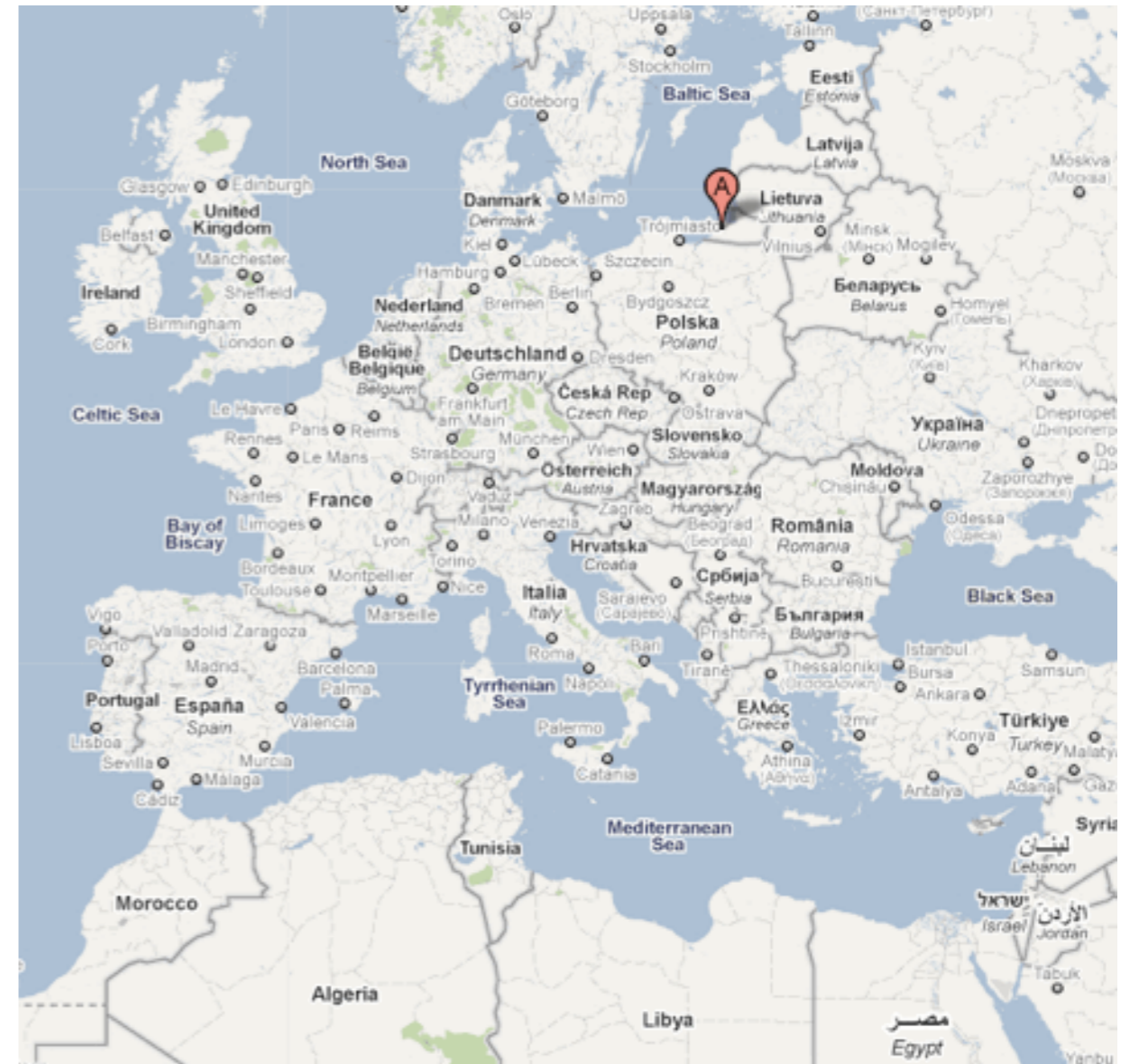


Digression...

On the shores of the Baltic Sea wedged between Lithuania and Poland is a region of Russia known as the [Kaliningrad Oblast](#).

The city of [Kaliningrad](#) is, by all accounts, a bleak industrial port with shoddy grey apartment buildings built hastily after World War II, when the city had been obliterated first by Allied bombers and later by the invading Russian forces.

Little remains of the beautiful Prussian city of [Königsberg](#), as it was formerly known.



Digression...

This is sad not only for lovers of architecture, but also for nostalgic mathematicians:

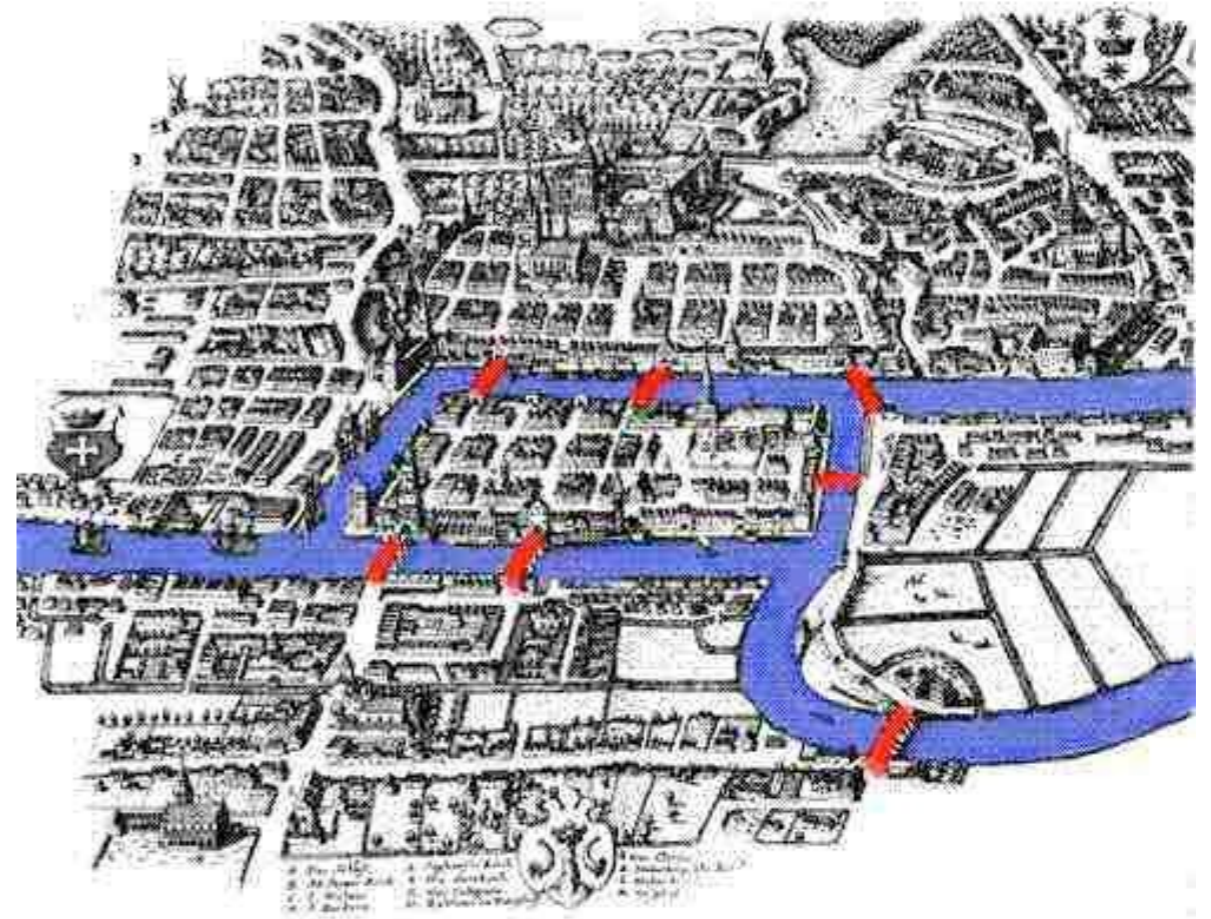
it was thanks to the layout of 18th century Königsberg that [Leonhard Euler](#) answered a puzzle which eventually contributed to two new areas of maths known as [topology](#) and [graph theory](#).



Digression...

Königsberg was built on the bank of the river Pregel.
Seven bridges connected two islands and the banks of the river (see map).

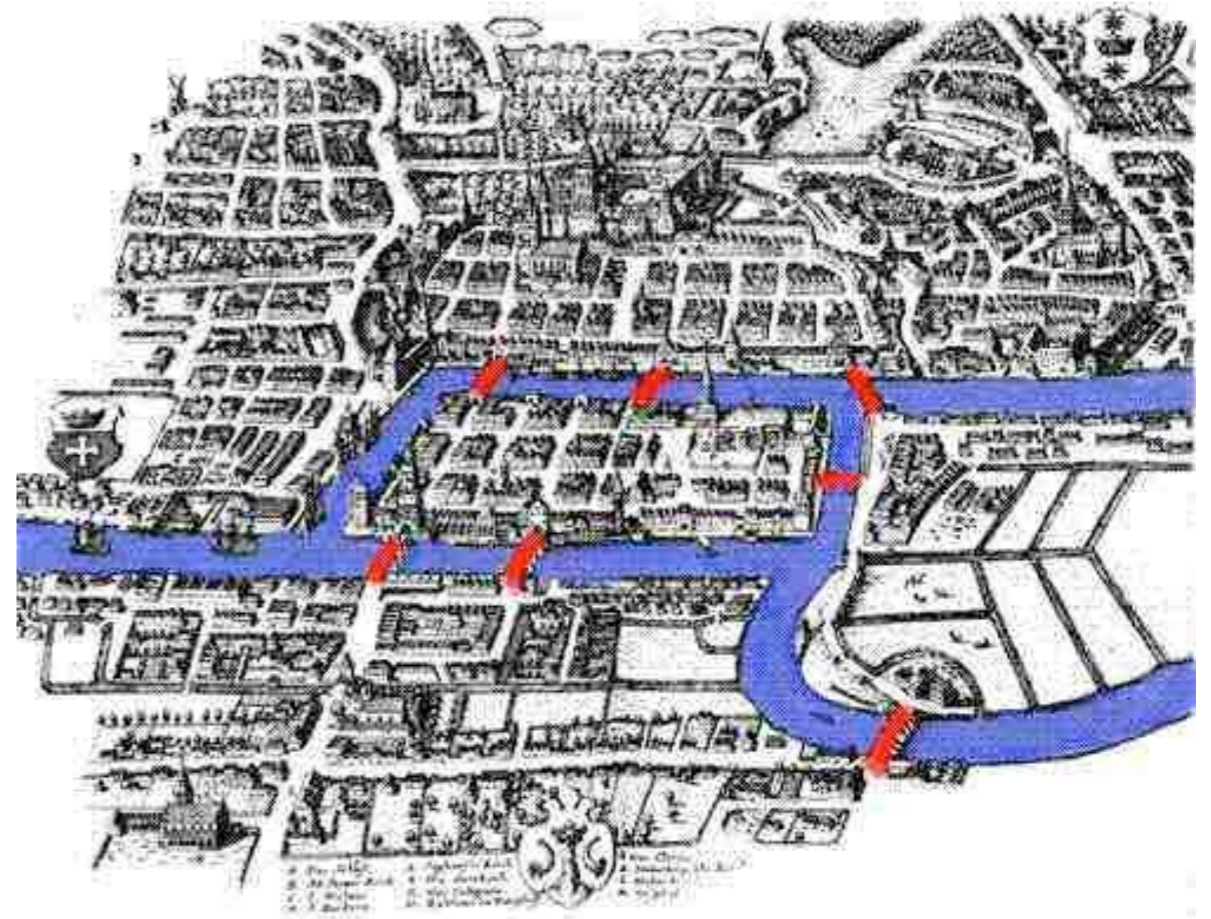
A popular pastime of the residents was to try to **cross all the bridges in one complete circuit** (without crossing any of the bridges more than once).



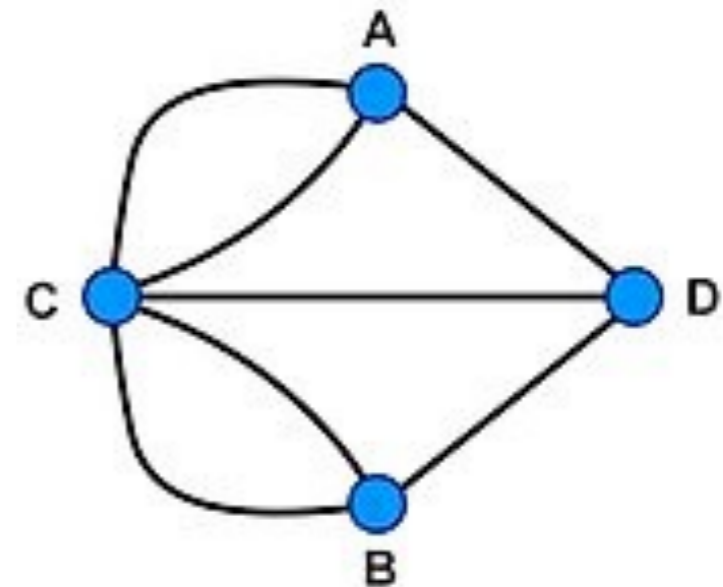
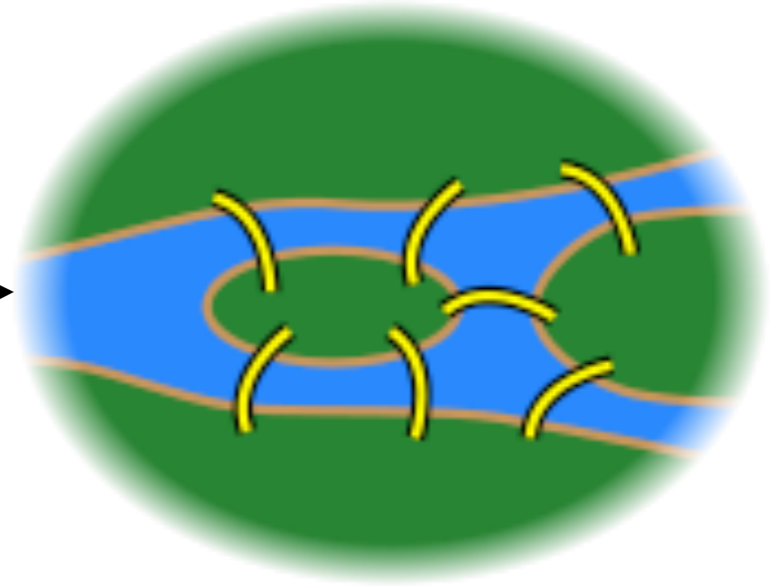
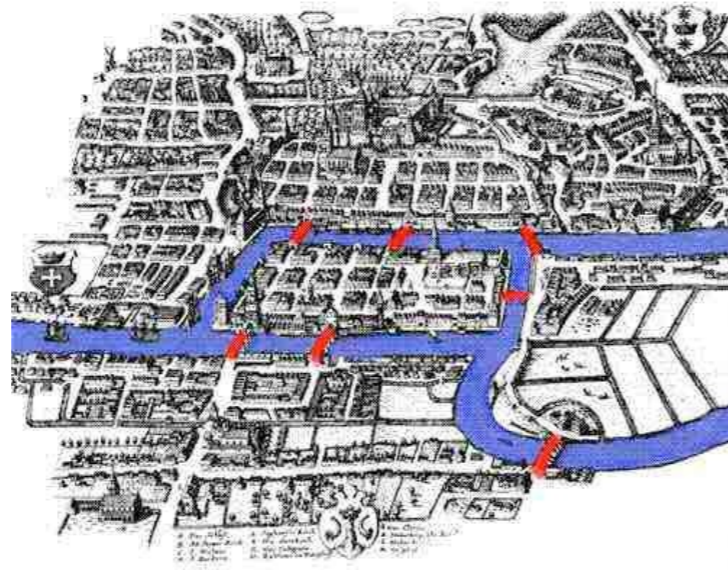
Digression...

A seemingly simple task was more than tricky...

Nobody had been able to find a solution to the puzzle when Euler first heard of it and, intrigued by this, he set about **proving** that **no solution was possible!**



Digression...



In 1736, Euler analysed the problem by converting the map into a more abstract diagram... and then into a **graph** (a formal model):

areas of land separated by the river were turned into points, which he labelled with capital letters. Modern graph theorists call these **vertices** or **nodes**.

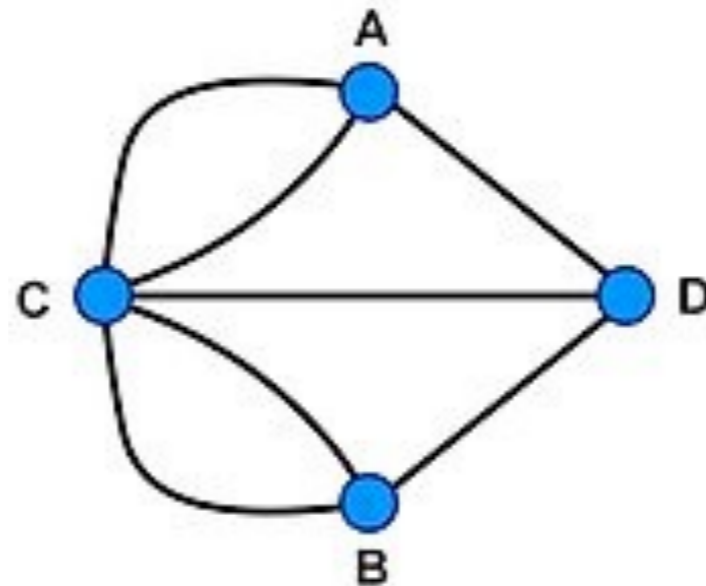
The bridges became **arcs** between nodes.

Digression...

Modeling activities require several steps of abstraction that must **preserve the set of solutions**: in other words the abstractions must preserve the topology of the problem.

Original problem: seven bridges of Königsberg

Graph problem: redrawing this picture without retracing any line and without picking your pencil up off the paper

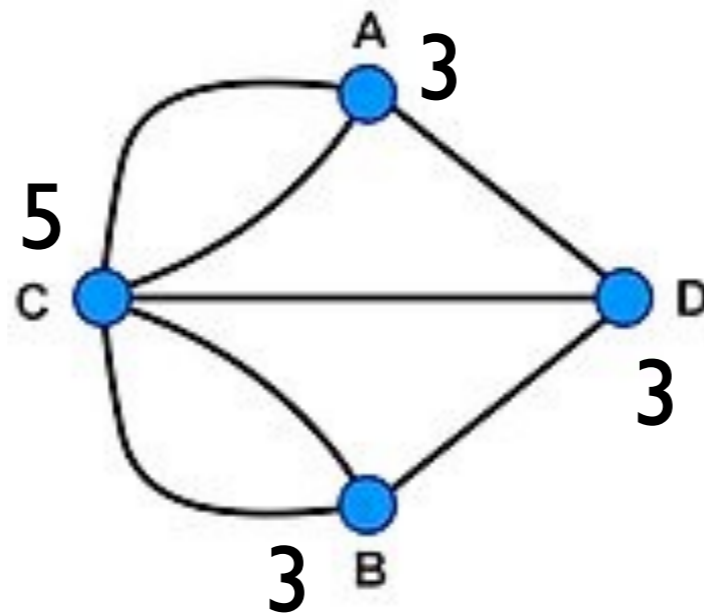


Generalized problem: given a **connected** graph, find a **circuit** that visits every edge precisely once, **if it exists**.

Digression...

Informal reasoning:

All the vertices in the above picture have an odd number of arcs connected to them.



Digression...

Informal reasoning:

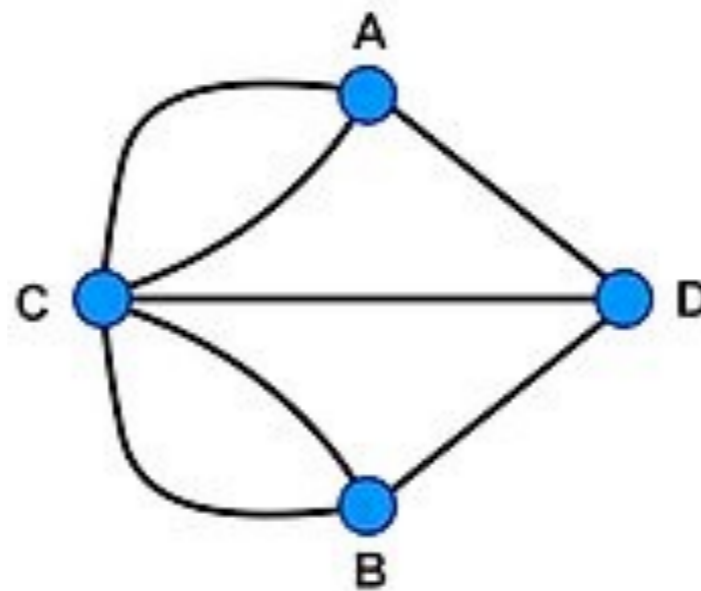
All the vertices in the above picture have an odd number of arcs connected to them.

Take one of these vertices, say D, and start trying to trace the figure out without picking up your pencil: then two arcs are left from/to D.

Next time you arrive in D, one arc will be left, and when you will leave D, no arc from/to it will be left!

Analogously for A, B, C.

No circuit possible!



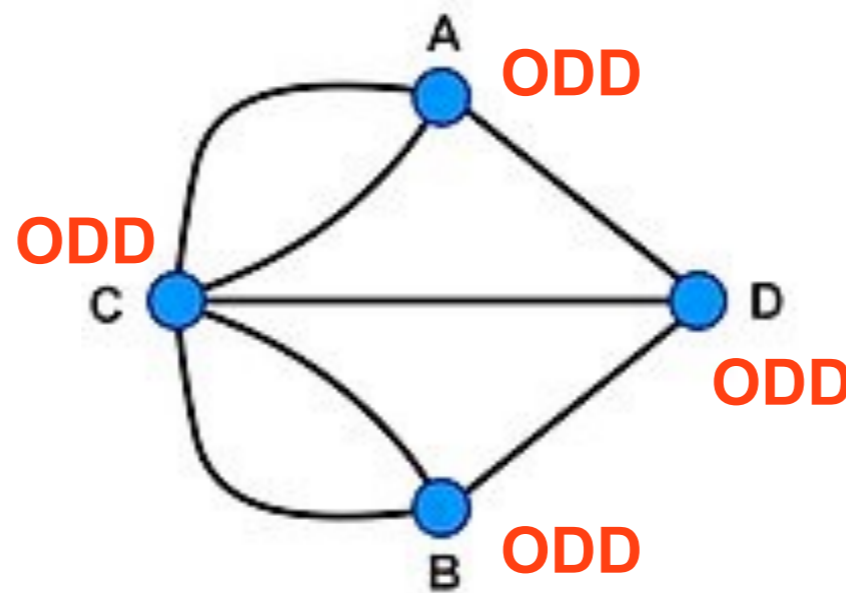
Digression...

Formal reasoning:

Definition: An **Eulerian path** is a continuous path that passes through every arc once and only once. It is a **circuit** if it ends in the same vertex where it starts.

Definition: A vertex is called **odd** if it has an odd number of arcs leading to it, otherwise it is called **even**. The number of arcs attached to node v is called **degree** of v .

Theorem: A (connected) graph G contains an Eulerian circuit if and only if the degree of each vertex is even.



Digression...

Proof of necessity: (existence of Eulerian circuit implies any vertex has even degree)

Suppose G contains an Eulerian circuit C .

Then, for any choice of vertex v , C contains all the edges that are adjacent to v .

Furthermore, as we traverse along C , we must enter and leave v the same number of times, and it follows that v must be even.

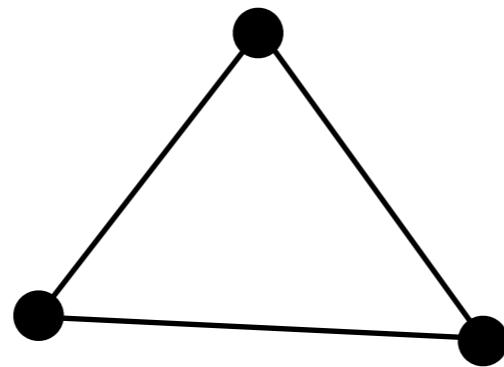
While this proof of necessity was given by Euler, the proof of converse is not stated in his paper.

It is not until 1873 (137 years later) when a young German mathematician, [Carl Hierholzer](#) published the proof of sufficiency.

Digression...

Proof of sufficiency: (by induction on the numbers of arcs)

Base case: the smallest possible number of edges is 3 (i.e. a triangle) and the graph trivially contains an Eulerian circuit.



Digression...

Proof of sufficiency: (by induction on the numbers of arcs)

Inductive case:

Inductive hypothesis: Let us assume that any connected graph H that contains k or less than k arcs and such that every vertex of H has even degree, contains an Eulerian circuit.

Now, let G be a graph with $k + 1$ edges, and every vertex has an even degree. We want to prove G has an Eulerian circuit

Since there is no odd degree vertex, G cannot be a tree (no leaves). Thus, G must contain at least one cycle C .

...

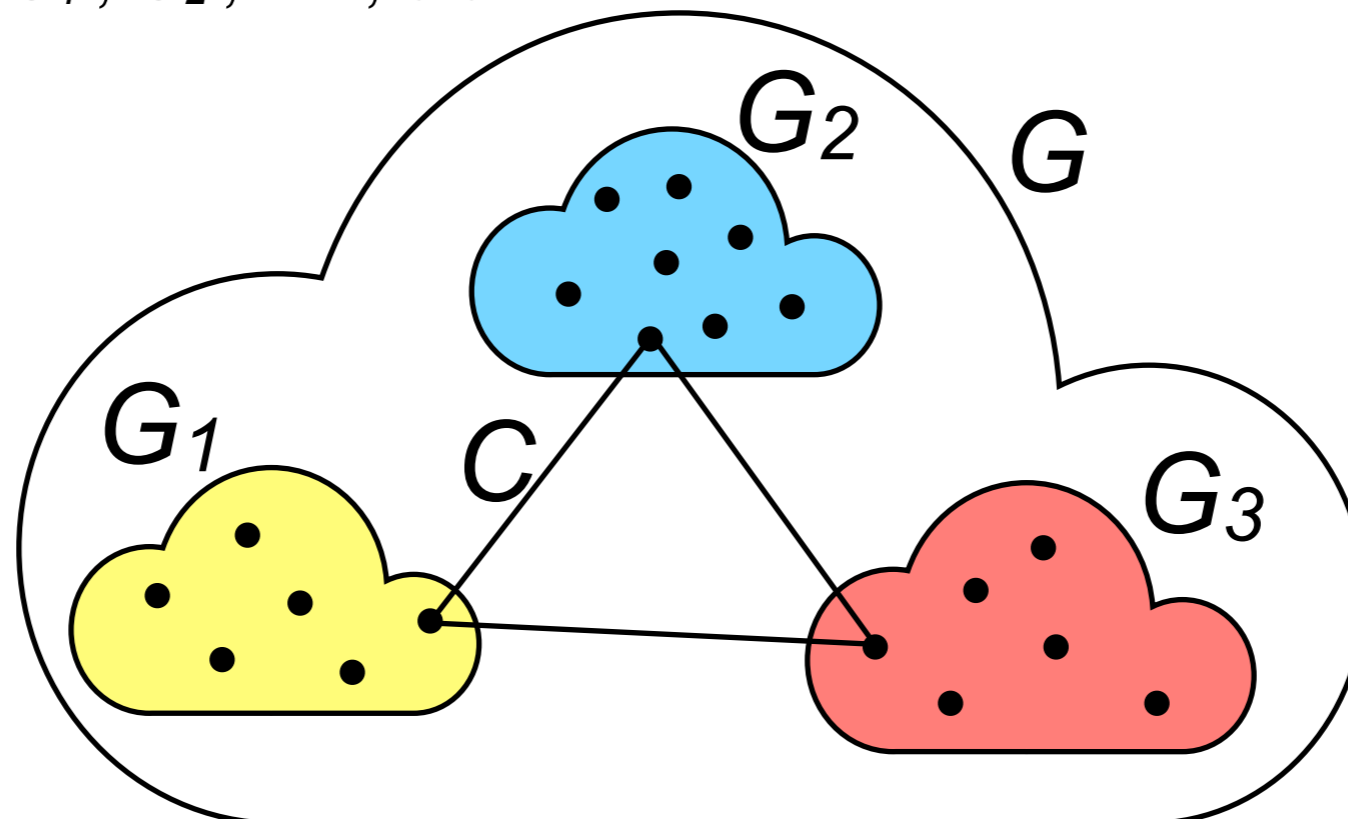
Digression...

Proof of sufficiency: (by induction on the numbers of arcs, continued)

...

Now, remove the edges of C from G , and consider the remaining graph G' .

Since removing C from G may disconnect the graph, G' is a collection of connected components, namely G_1, G_2, \dots , etc.



...

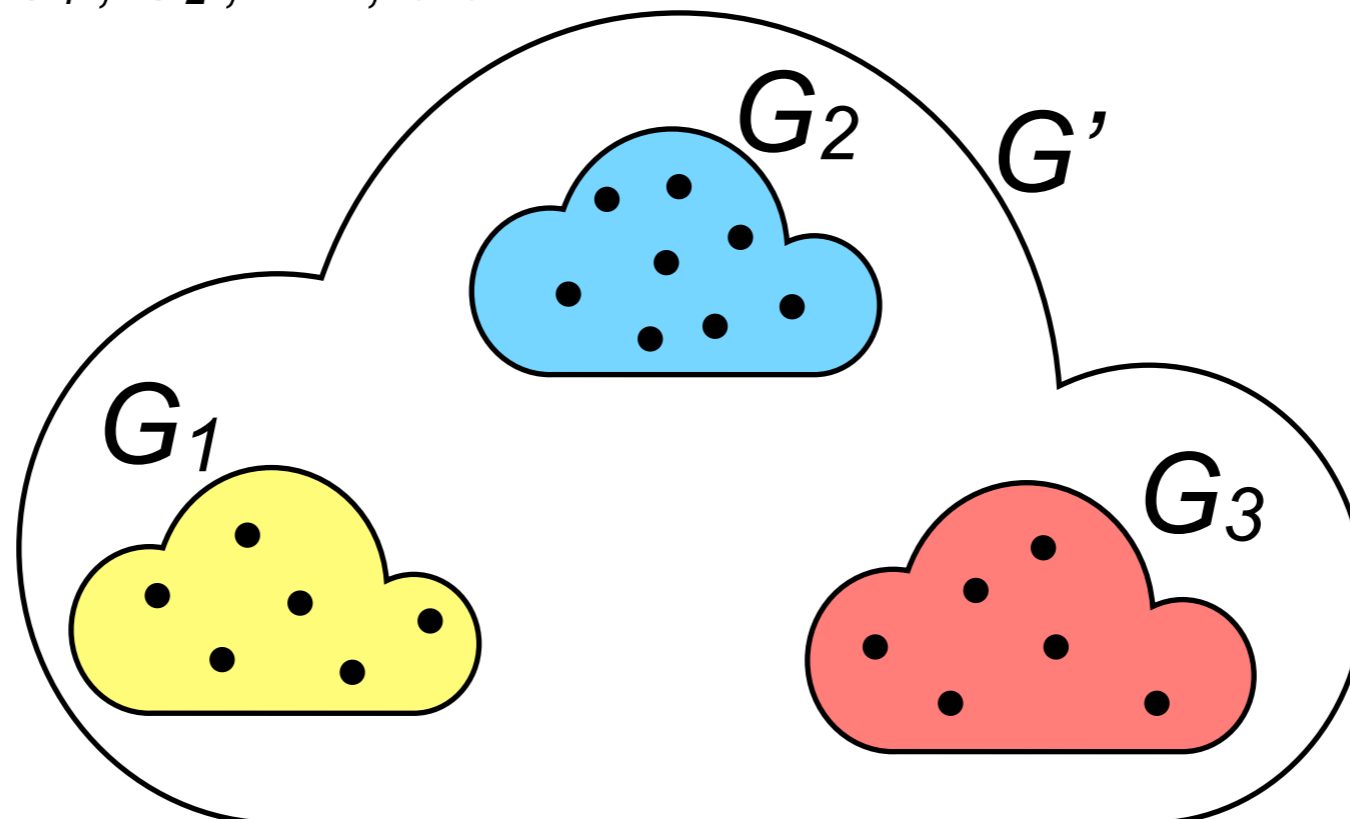
Digression...

Proof of sufficiency: (by induction on the numbers of arcs, continued)

...

Now, remove the edges of C from G , and consider the remaining graph G' .

Since removing C from G may disconnect the graph, G' is a collection of connected components, namely G_1, G_2, \dots , etc.



...

Digression...

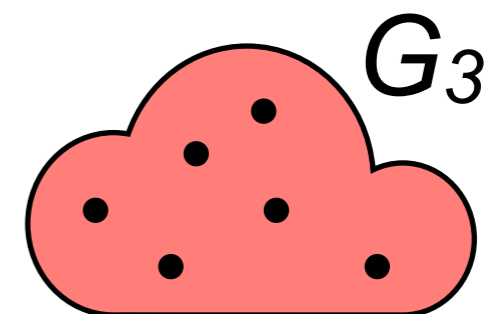
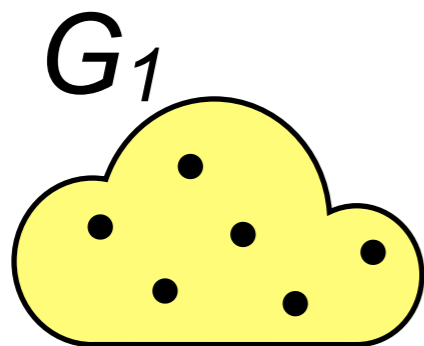
Proof of sufficiency: (by induction on the numbers of arcs, continued)

...

Furthermore, when the edges in C are removed from G , each vertex loses even number of adjacent edges. Thus, the parity of each vertex is unchanged in G' .

It follows that, for each connected component of G' , every vertex has an even degree.

Therefore, by the induction hypothesis, each of G_1, G_2, \dots has its own Eulerian circuit, namely C_1, C_2, \dots , etc.



...

Digression...

Proof of sufficiency: (by induction on the numbers of arcs, continued)

...

We can now build an Eulerian circuit for G .

Pick an arbitrary vertex v from C .

Traverse along C until we reach a vertex v_i that belongs to one of the connected components G_i .

Then, traverse along its Eulerian circuit C_i until we traverse all the edges of C_i .

We are now back at v_i , and so we can continue on along C .

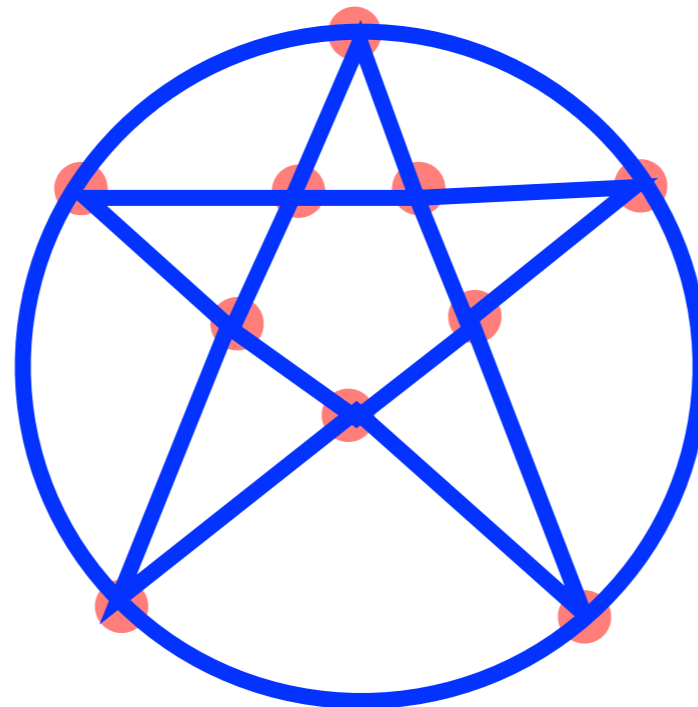
In the end, we shall return back to the first starting vertex v , after visiting every edge exactly once.

Digression...

The theorem, as such, is only an existential statement.

If the necessary and sufficient condition is satisfied, we wish to find an Eulerian circuit.

The inductive proof naturally gives an algorithm to construct Eulerian circuits:
recursively find a cycle, and then remove the edges of the cycle.

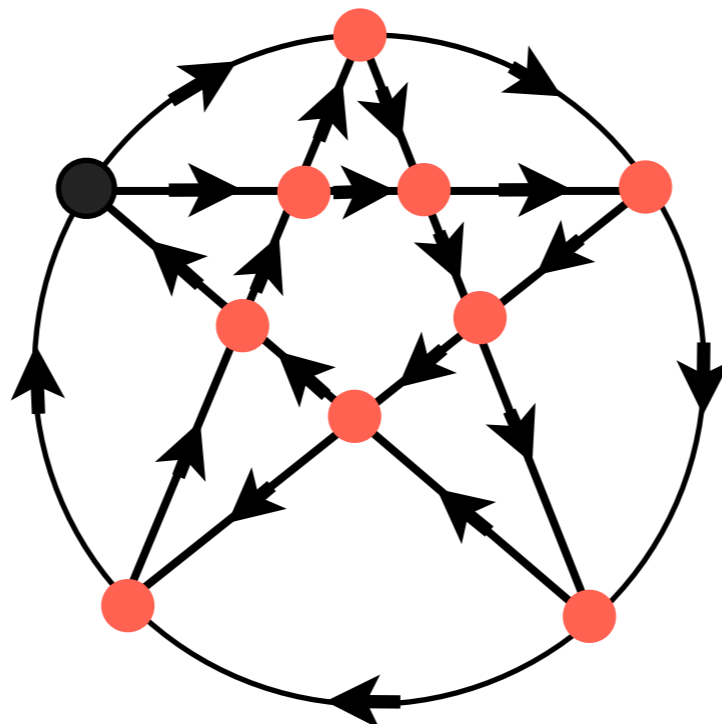


Digression...

The theorem, as such, is only an existential statement.

If the necessary and sufficient condition is satisfied, we wish to find an Eulerian circuit.

The inductive proof naturally gives an algorithm to construct Eulerian circuits:
recursively find a cycle, and then remove the edges of the cycle.



Digression...

Theorem: A graph contains an **Eulerian path** if and only if there are 0 or 2 odd vertices.

Proof.

Suppose a graph G contains an Eulerian path P .

Then, for every vertex v , P must enter and leave v the same number of times, except when it is either the starting vertex or the final vertex of P .

When the starting and final vertices are distinct, there are precisely 2 odd degree vertices.

When these two vertices coincide, there is no odd degree vertex.

Conversely, suppose G contains 2 odd degree vertex u and v .

(The case where G has no odd degree vertex is shown in the previous Theorem.)

Then, temporarily add a dummy edge (u, v) to G .

Now the modified graph contains no odd degree vertex.

By the previous Theorem, this graph contains an Eulerian circuit C that includes (u, v) .

Remove (u, v) from C , and now we have an Eulerian path where u and v serve as initial and final vertices.

Digression...

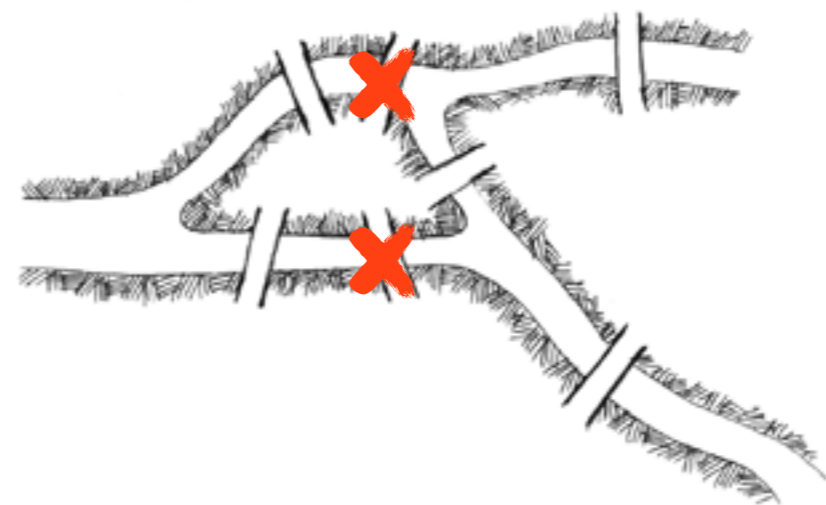
In the late 19th century an eighth bridge was built (see map).

As a result Königsberg had been Eulerised!

Exercise: prove that an Eulerian path can be found (but not a circuit)

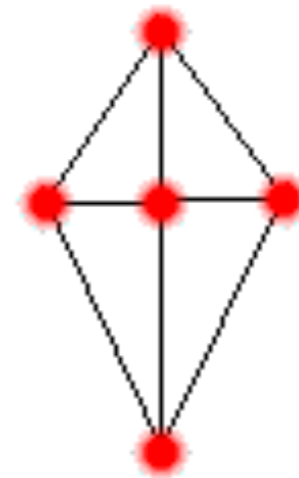
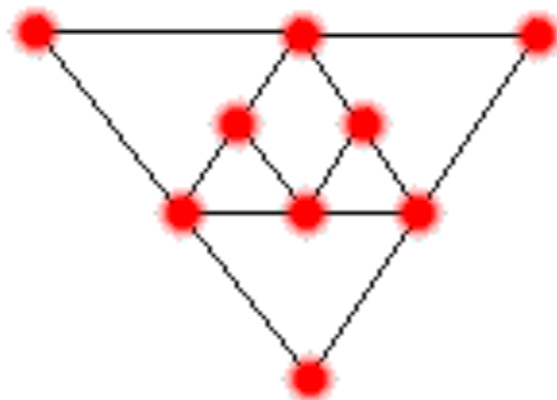
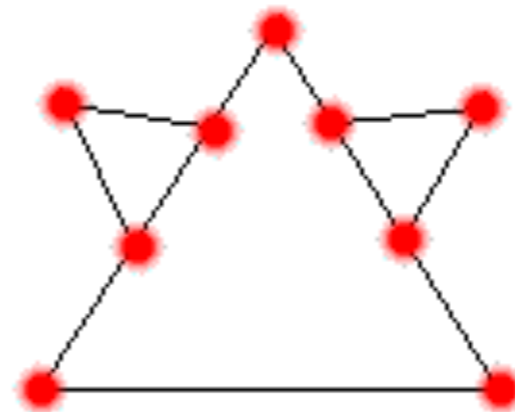
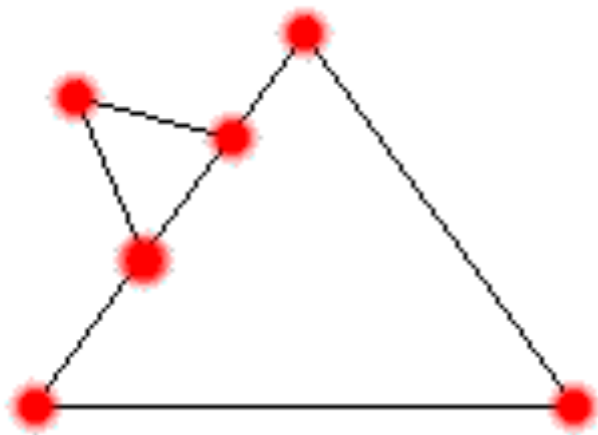


Sadly, in 1944 air raids obliterated most of the bridges. However, from the maps made available since, it appears that five bridges crossing were rebuilt in such a way that Kaliningrad was Eulerised once again!



Exercise: prove that an Eulerian path can be found (but not a circuit)

Digression...



Exercises: find Eulerian paths/circuits in the graphs above or prove that they cannot exist.

Lessons learned

- Concrete instance of the problem
- Abstract modeling and generalization
- Visual notation, informal, intuitive
- Mathematical notation, rigorous, precise
- Solutions from formal reasoning, proofs
- Implementation and application to concrete instances

Yet to learn

- Formal models used in prescriptive manner
- Correctness by design
- Separation of concerns
- Model discovery