Packaged Software and Data Modelling –
The Surprising Reasons Behind Implementation Failures

Adept Events Data Warehousing and Business Intelligence Summit
02 april 2025 in Utrecht NL

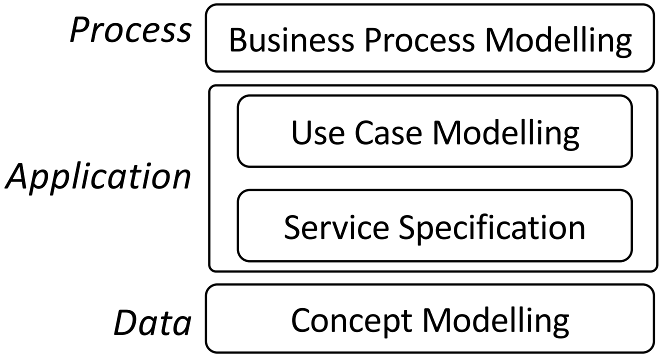
Alec Sharp
Senior Consultant
Clariteq Systems Consulting Ltd.
West Vancouver, BC, Canada
asharp@clariteq.com
www.clariteq.com

Speaker background…

Alec Sharp, Clariteq Systems Consulting – *asharp@clariteq.com*

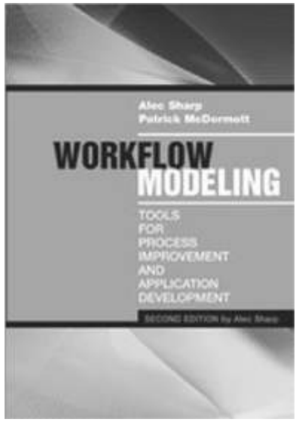
- 40+ years experience as an independent consultant:
  - Business Process Change – discover, model, analyse, and design/redesign processes
  - Application Requirements Specification
  - Data Modelling and Management

    +
  - Facilitation & Organisational Change
  - Project Recovery



- Consulting, teaching, speaking globally (pre-pandemic)
- Awarded DAMA's global Professional Achievement Award for contributions to "human-friendly" data modelling
- Author of "Workflow Modeling"
  - best-selling book on process modelling & improvement
  - second edition – a complete re-write

Check out the nice reviews on Amazon - http://amzn.to/dHun1o


Alec Sharp
Patrick McDermott
WORKFLOW MODELING
TOOLS FOR PROCESS IMPROVEMENT AND APPLICATION DEVELOPMENT
SECOND EDITION by Alec Sharp

Project recovery

Organisations spend hugely on packaged solutions (COTS)
(and custom solutions) often with negative outcomes:

- High-tech manufacturer: US$500M -> US$1B -> US$1.6B
- Famous research university: US$80M
- Big Box Retailer: US$2.4B
- Canadian Government Phoenix Pay System – CA$2.6B?

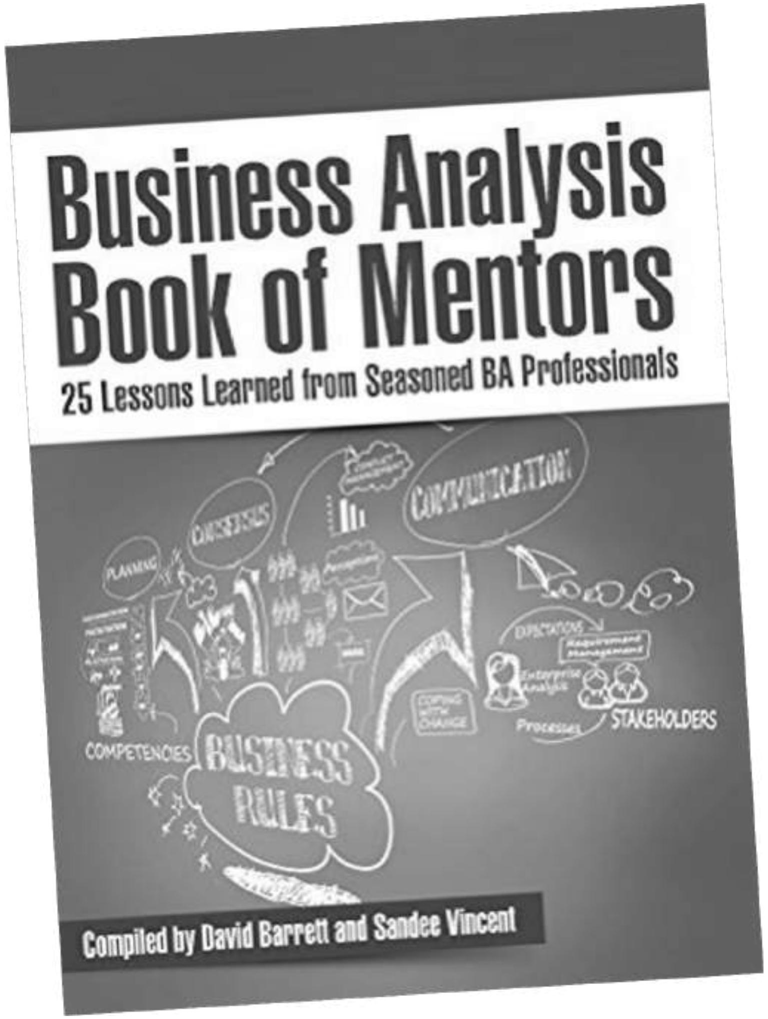
I’m often called in for "project recovery"

Why do things go so badly wrong? Organisations dive quickly into detailed requirements and experience:

- a failure to understand *end-to-end* business processes
- a concept model (data model) mismatch
- *management (idiotically) decreeing “no process change” – solution must replicate existing practices*

Getting to the essence is always part of the solution – "getting out of the weeds"

My chapter in the "BA Book of Mentors"

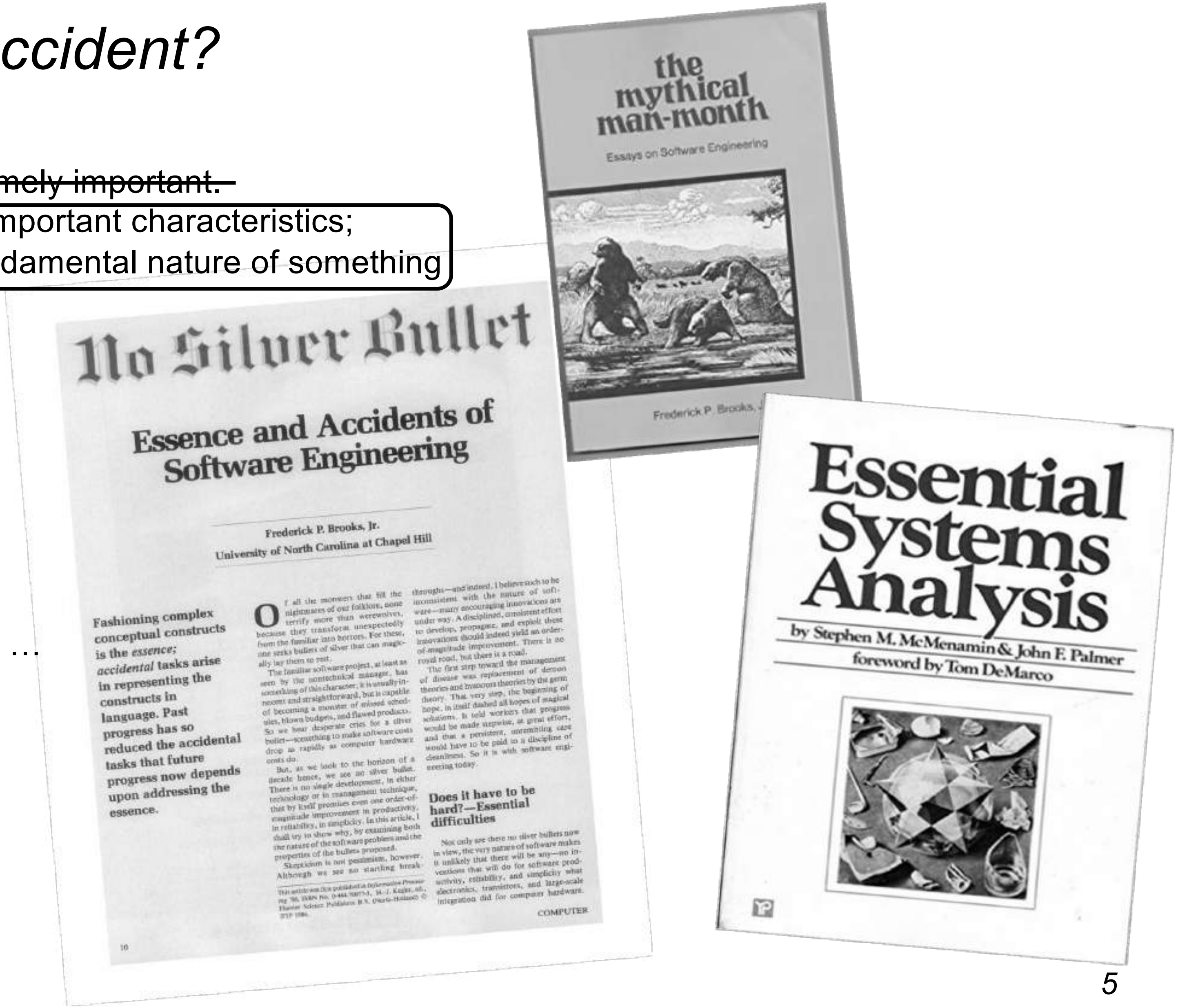

Business Analysis Book of Mentors
25 Lessons Learned from Seasoned BA Professionals
Compiled by David Barrett and Sandee Vincent

The premise of the book:

- 25 experienced BAs from around the world would each write a chapter on "The Most Important Lesson I Learned in my BA Career.'
- I knew mine instantly – separate the "what" from the "who, how, and why"
- In other words, separate the "essence" from the "accident"

Essence and Accident?

Essential:
1) ~~absolutely necessary; extremely important.~~
2) something's basic or most important characteristics;
the intrinsic, inherent, or fundamental nature of something



No Silver Bullet

Essence and Accidents of Software Engineering

Frederick P. Brooks, Jr.
University of North Carolina at Chapel Hill

Fashioning complex conceptual constructs is the *essence*; *accidental* tasks arise in representing the constructs in language. Past progress has so reduced the accidental tasks that future progress now depends upon addressing the essence.

Of all the monsters that fill the nightmares of our folklore, none terrify more than werewolves, because they transform unexpectedly from the familiar into horrors. For these, one seeks bullets of silver that can magically lay them to rest.

The familiar software project, at least as seen by the nontechnical manager, has something of this character; it is usually innocent and straightforward, but is capable of becoming a monster of missed schedules, blown budgets, and flawed products. So we hear desperate cries for a silver bullet—something to make software costs drop as rapidly as computer hardware costs do.

But, as we look to the horizon of a decade hence, we see no silver bullet. There is no single development, in either technology or in management technique, that by itself promises even one order-of-magnitude improvement in productivity, in reliability, in simplicity. In this article, I shall try to show why, by examining both the nature of the software problem and the properties of the bullets proposed.

Skepticism is not pessimism, however. Although we see no startling breakthroughs—and indeed, I believe such to be inconsistent with the nature of software—many encouraging innovations are under way. A disciplined, consistent effort to develop, propagate, and exploit these innovations should indeed yield an order-of-magnitude improvement. There is no royal road, but there is a road.

The first step toward the management of disease was replacement of demon theories and humours theories by the germ theory. That very step, the beginning of hope, in itself dashed all hopes of magical solutions. It told workers that progress would be made stepwise, at great effort, and that a persistent, unremitting care would have to be paid to a discipline of cleanliness. So it is with software engineering today.

Does it have to be hard?—Essential difficulties

Not only are there no silver bullets now in view, the very nature of software makes it unlikely that there will be any—no inventions that will do for software productivity, reliability, and simplicity what electronics, transistors, and large-scale integration did for computer hardware.

This article was first published in Information Processing '86, ISBN No. 0-444-70077-3, H.-J. Kugler, ed., Elsevier Science Publishers B.V. (North-Holland) © IFIP 1986.

10

COMPUTER

the mythical man-month
Essays on Software Engineering
Frederick P. Brooks, J

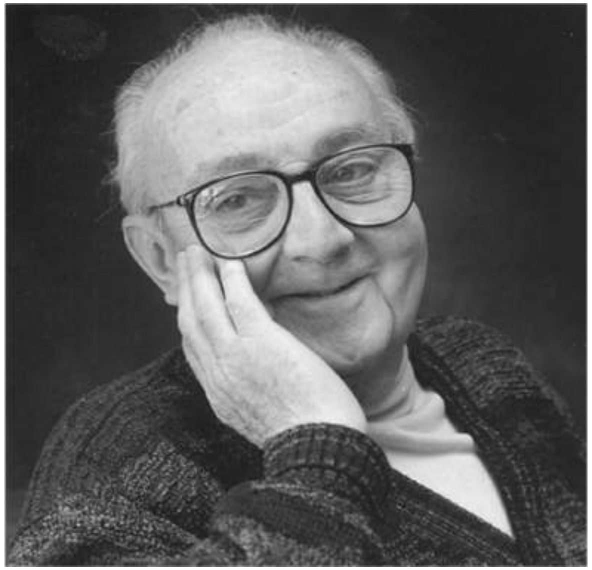
Essential Systems Analysis
by Stephen M. McMenamin & John F. Palmer
foreword by Tom DeMarco

Cup:
The *essential* characteristics:
a round, handheld container
for drinking from.
What it is.

The *accidental* characteristics:
ceramic vs. bamboo, handle or not, …
How it is designed or made.

"All models are wrong, but some are useful."

“... all models are approximations. Essentially, *all models are wrong, but some are useful*. However, the approximate nature of the model must always be borne in mind…”

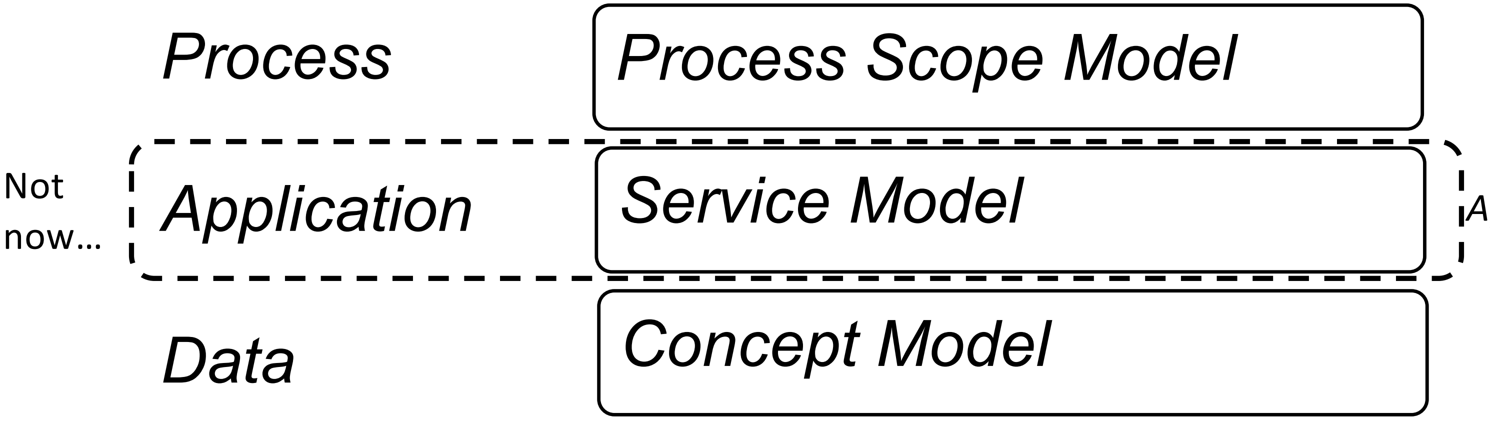


George E. P. Box
1919–2013

The Law of Parsimony:
Since all models are wrong,
the scientist cannot obtain a "correct" one
by excessive elaboration.

On the contrary, following William of Ockham (of "Occam's Razor" fame)
… seek an economical description ….
Just as the ability to devise simple but evocative models
is the signature of the great scientist,
so overelaboration … is often the mark of mediocrity.

Two especially useful (and simple!) essential models



Process

Process Scope Model

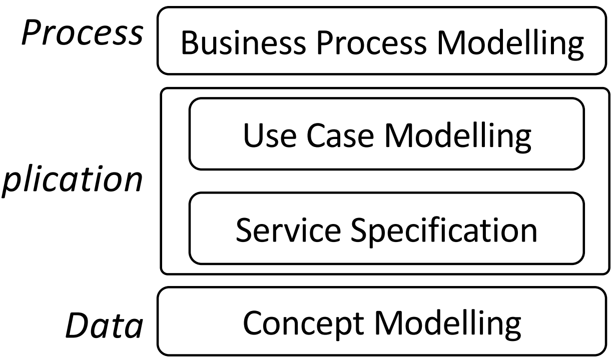
Not now…

Application

Service Model

Data

Concept Model



Process

Business Process Modelling

Application

Use Case Modelling

Service Specification

Data

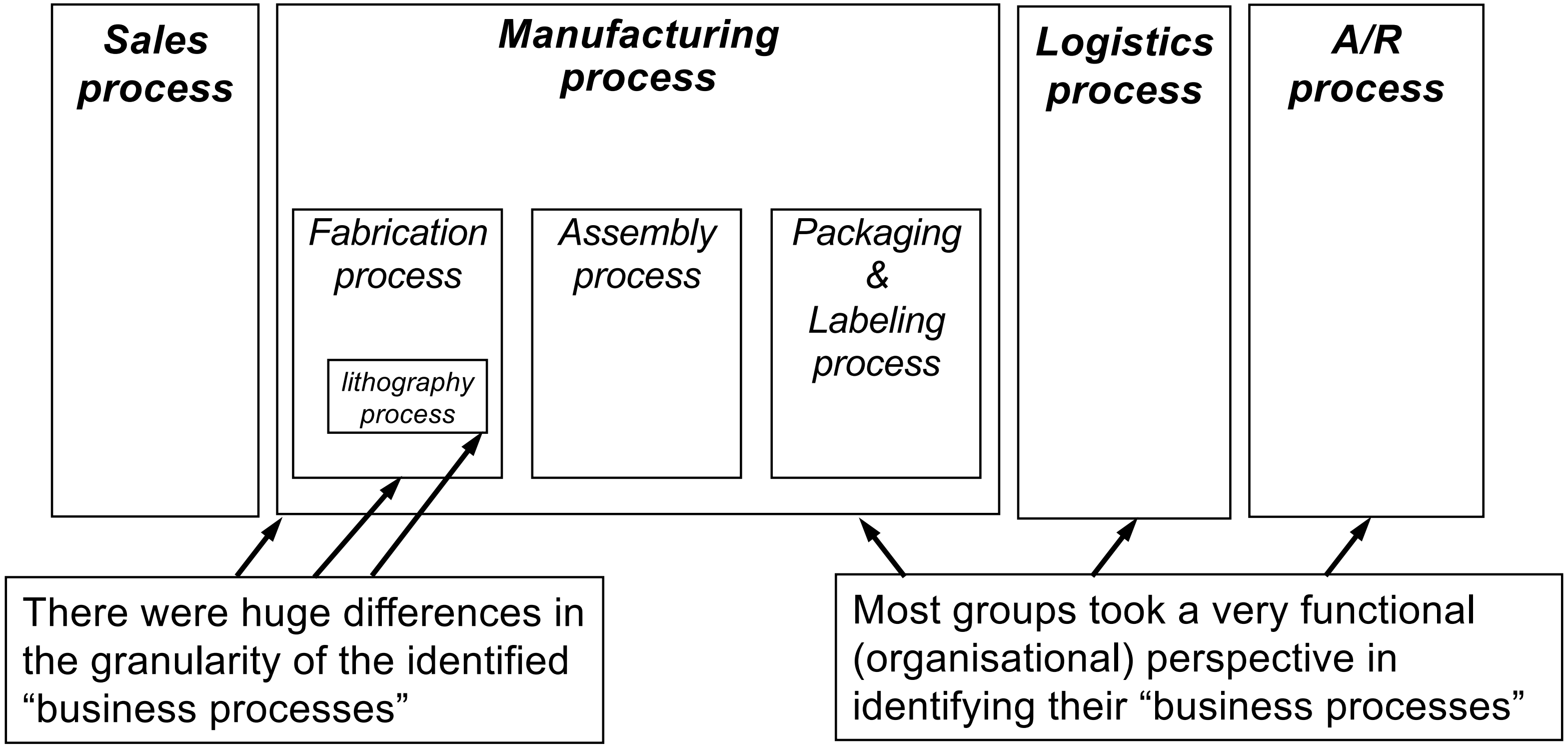
Concept Modelling

All are “essential” –
they show the essence of a subject – the *what* –
with no reference to *who, how, why, etc.*

Of course, *who, how, when, where, and why* are important – *later!*

A real life (and expensive!) example

As part of a massive system implementation, a global manufacturer identified the *business processes* that were expected to improve:

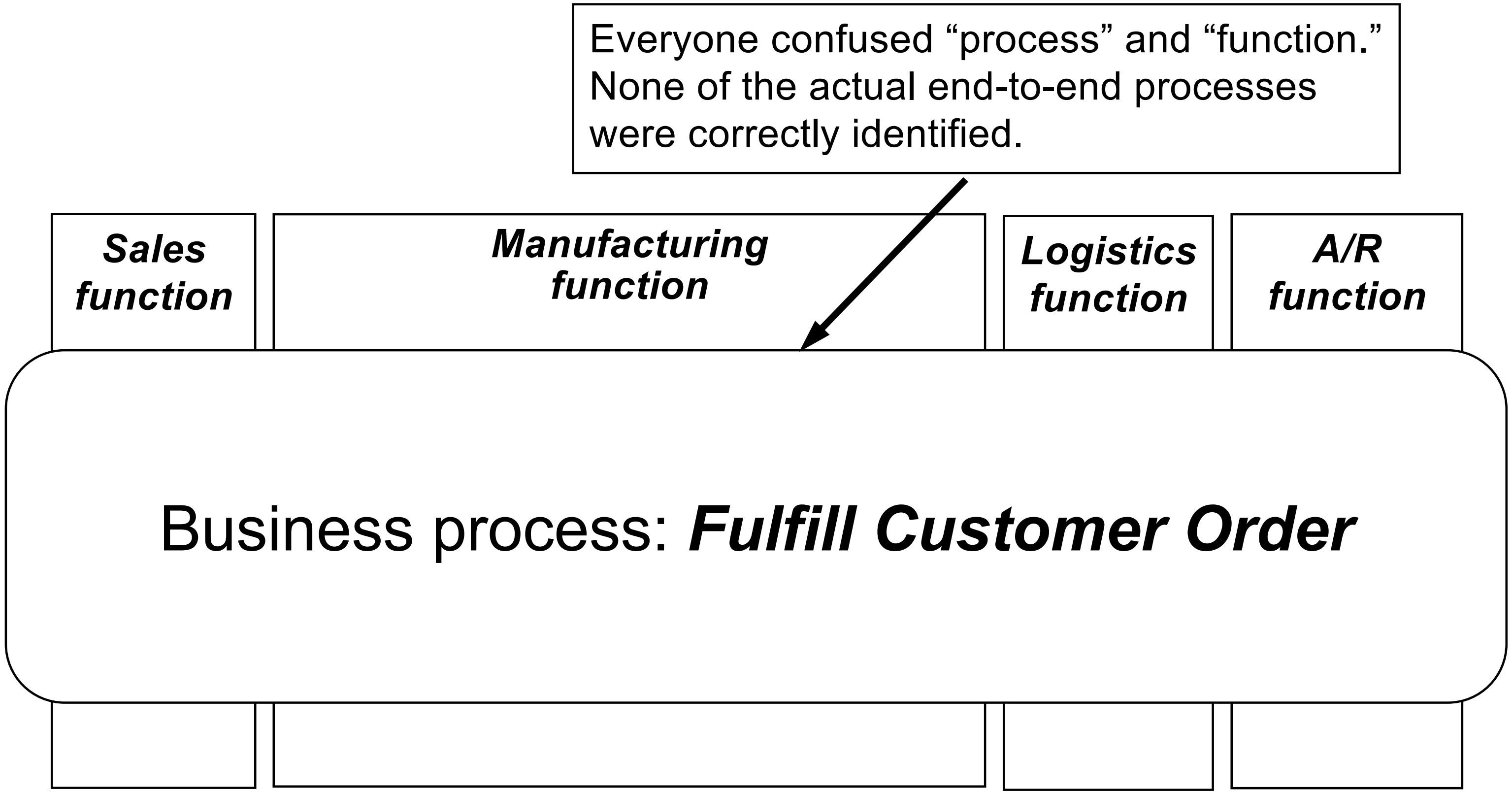


There were huge differences in the granularity of the identified "business processes"

Most groups took a very functional (organisational) perspective in identifying their "business processes"

The problem? *These aren't processes – they're functions!*

The “real” business processes were missed



Everyone confused “process” and “function.” None of the actual end-to-end processes were correctly identified.

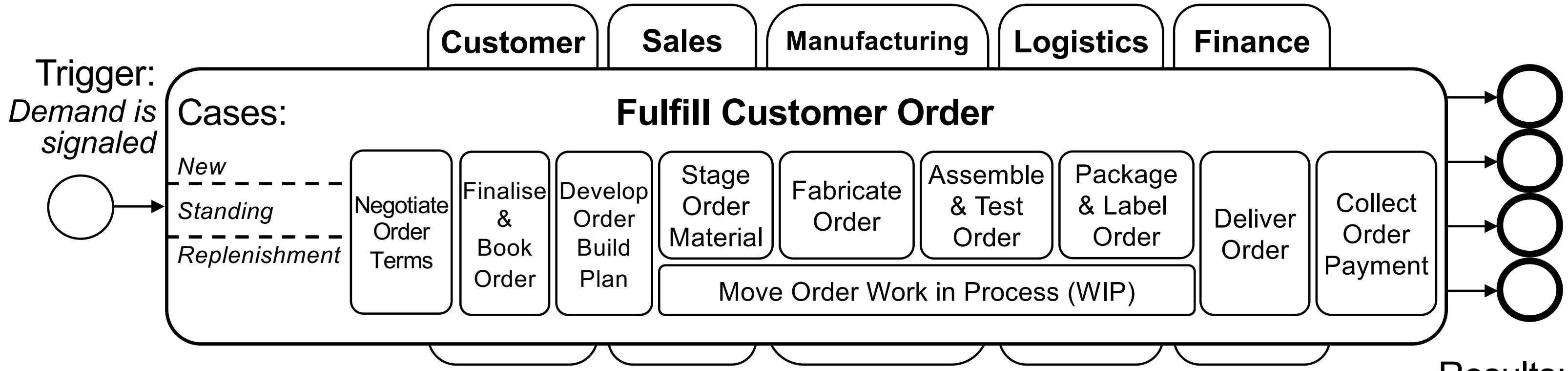
Sales function
Manufacturing function
Logistics function
A/R function
Business process: ***Fulfill Customer Order***

“Business Process” =
end-to-end, cross-functional, business process.

“Larger” than people think – from *initial* trigger to *final* results

Get out of the weeds with a Process Scope Model

I build a *Process Scope Model* & a *Process Summary Chart* on ~100% of *Project Recovery* assignments -



Trigger:
Demand is signaled

Customer | Sales | Manufacturing | Logistics | Finance

Fulfill Customer Order

Cases:
New
Standing
Replenishment

Negotiate Order Terms

Finalise & Book Order

Develop Order Build Plan

Stage Order Material

Fabricate Order

Assemble & Test Order

Package & Label Order

Move Order Work in Process (WIP)

Deliver Order

Collect Order Payment

Results:

Customer:
Goods received, tested, & accepted

Owner:
Payment received

Performer:
Commission credited

Industry Association:
Order stats reported

"TRAC" –

*1 – **T**riggering event or events*

*2 – **R**esults: final outputs*

- result(s) received by the process' primary customer
- result(s) for other stakeholders (performers, owner, supplier, regulator, …)

*3 – **A**ctivities: ~5 – 7 phases, milestones, major activities, …*

- a phase achieves a significant intermediate result
- simply ask the participants for key milestones within the process

*4 – **C**ases*

- main variations, e.g. "*new* order" vs. "*standing* order"
- verb – *qualifier* – noun

essence** of the process **("what")

5 – Functions or Organisation Units

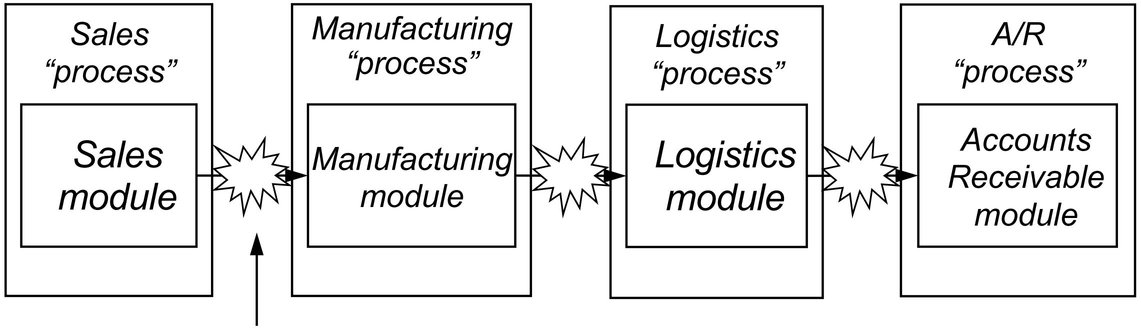
6 – Actors and responsibilities

7 – Systems, data sources, other mechanisms

as-is elements** of the process, for clarification **("who and how")

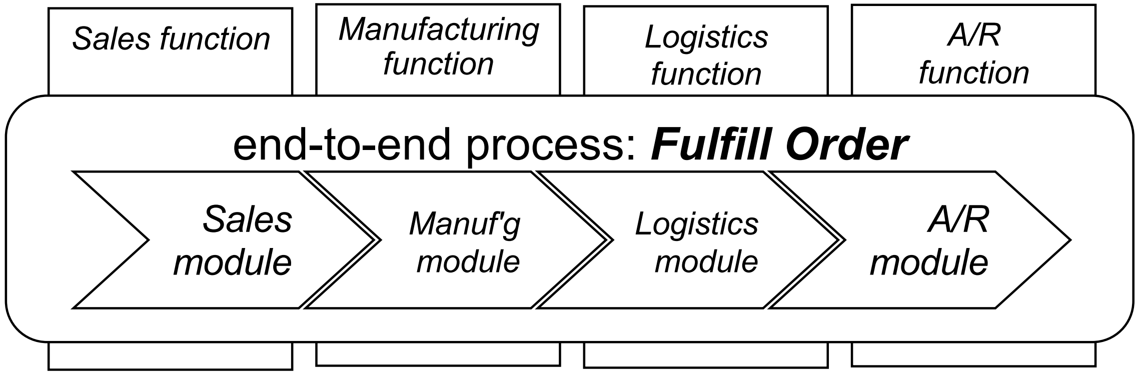
Impact of confusing function and process

Implementing SAP without clarity on "process":



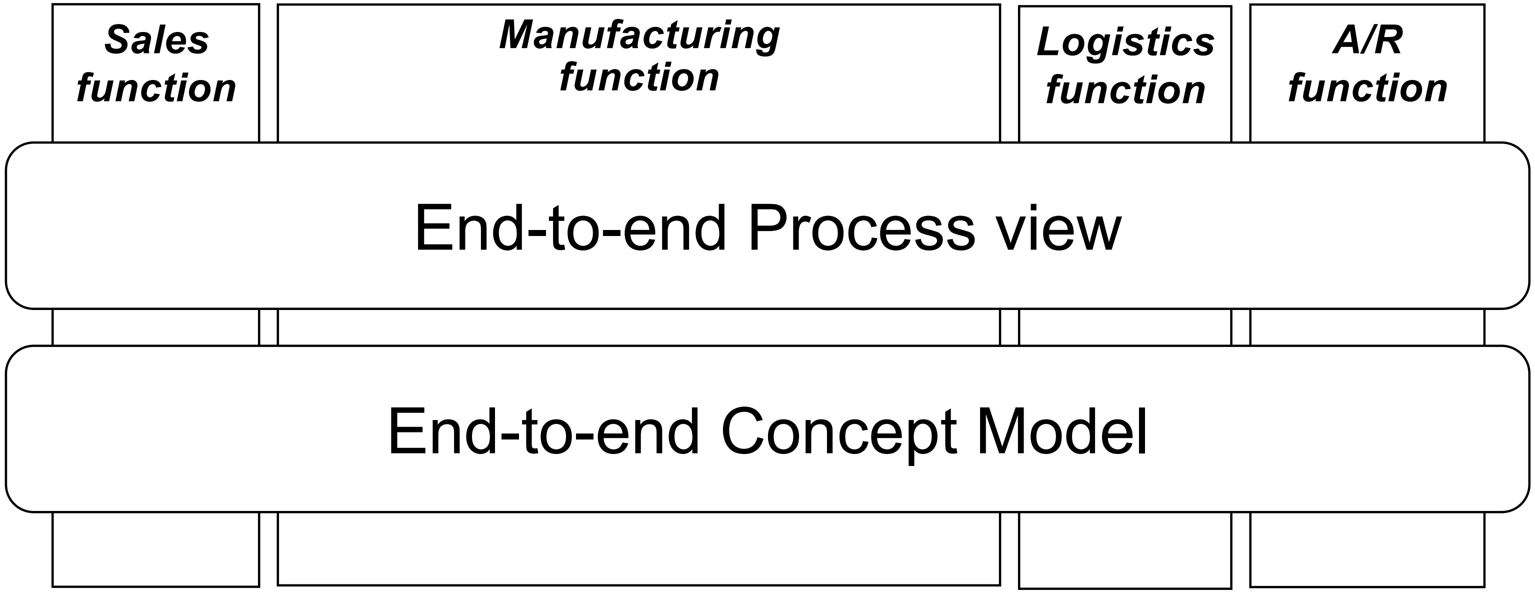
Conflicts: timing, coding, terminology, data formats, *performance targets*, …

SAP re-implemented in a process-driven configuration:



Same software, radically different outcomes

The end-to-end Concept Model was also crucial



Example: If you ignore the process and the data…

U.S. University implementing cloud-based Human Resources and Payroll systems from *the same vendor.*

- Total spend US$80M, nothing salvageable
- University leadership unamused
- I was brought in for “project recovery”

The situation

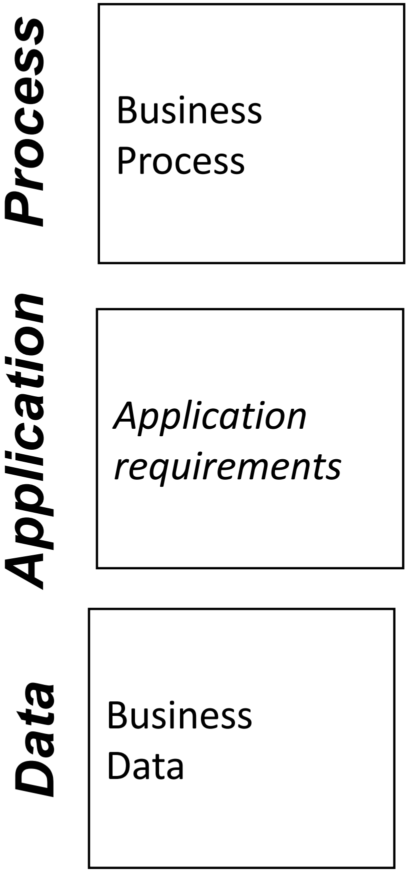
What we learned:

- Little time on “business process”
  - very generic / unrecognisable as “what we do”
  - team tires of this
- Zero time on “data” (no “concept model”)
- Management: "Get on with it – the vendor has seen it all before."
- 100+ programmers begin detailed configuration of *application rules and logic* – *“Straight to task.”*

My assignment –
take a large team through a process model
and data model-based approach –
run 4-day offsite in “The Capsule”
(we felt like astronauts)

A "Futuro" house – Finnish architect Matti Suuronen

Initial focus – too much on "requirements"

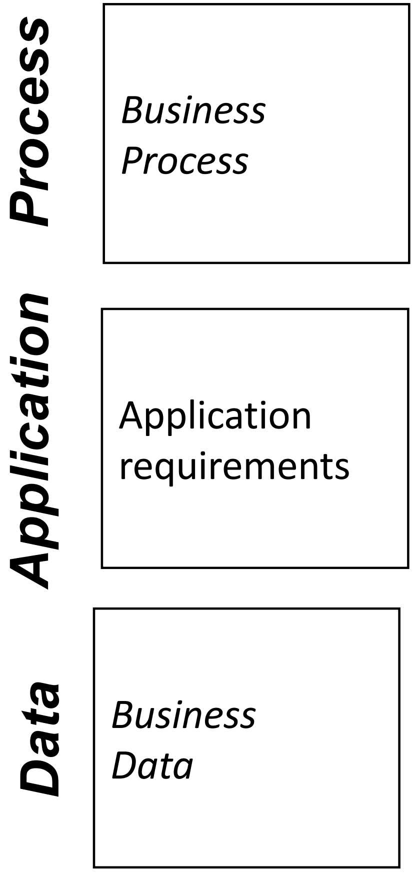


Over 100 developers coded detailed business rules and contract terms into

- Payroll Application
- HR Application

Note: university had over 35 labour unions with complex payroll and benefits policies/rules – ***no rethinking whatsoever!***

Remediation – focus on process and data



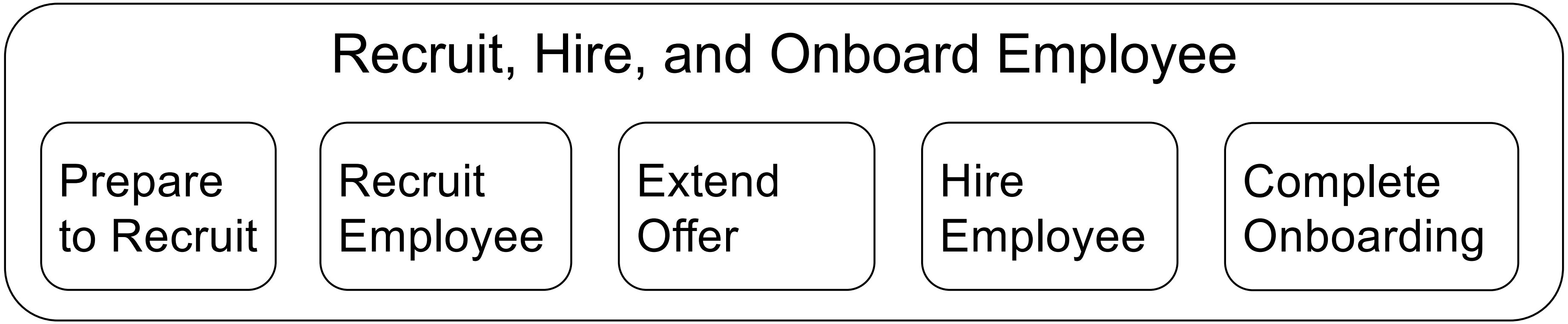
Identified, modelled, analysed, redesigned significant process – “Recruit, Hire, and Onboard Employee," the Case was "Tenure-Track Faculty”

- Developed scope model (invaluable!)
- Developed augmented scope model
- Assessed and redesigned based on “what”
- Built to-be scope model to “who – what – how” detail

Modelled seven critical concepts in data – “what do we mean by…”

- Supervisory-Organisational Hierarchy
- Position-Based Management
- Visible Application Workflow
- etc.

First, identify main phases in a Scope Model



Recruit, Hire, and Onboard Employee

Prepare to Recruit

Recruit Employee

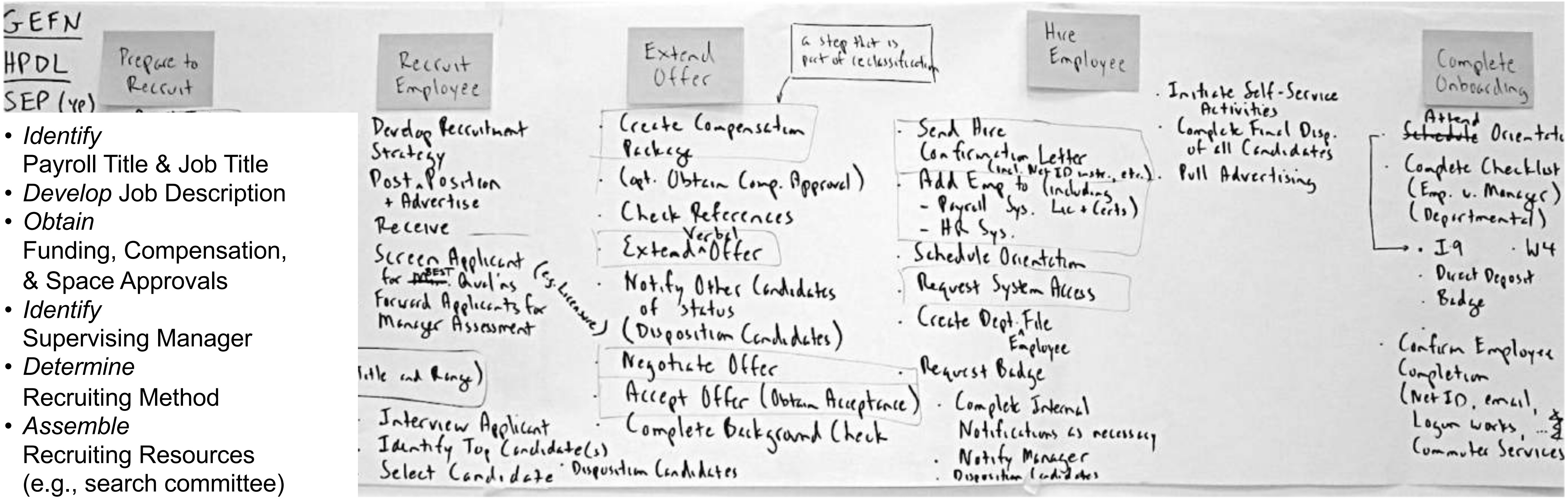
Extend Offer

Hire Employee

Complete Onboarding

Augmented Scope Model for the full process

Recruit, Hire, and Onboard Employee

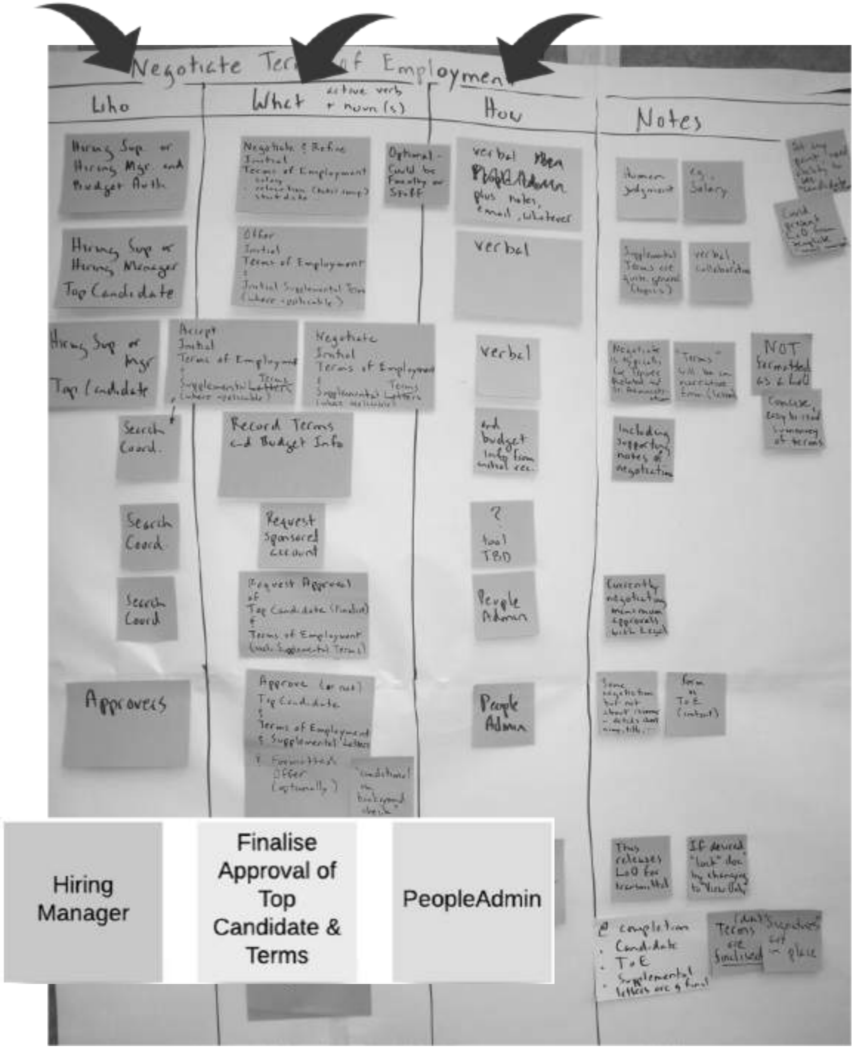

GEFN
HPDL
SEP (ye)
Prepare to Recruit
Recruit Employee
Develop Recruitment Strategy
Post Position + Advertise
Receive
Screen Applicant for BEST Qual'ns (e.g. Licensure)
Forward Applicants for Manager Assessment
Title and Range)
Interview Applicant
Identify Top Candidate(s)
Select Candidate
Disposition Candidates
Extend Offer
a step that is part of reclassification
Create Compensation Package
(opt. Obtain Comp. Approval)
Check References
Extend Verbal Offer
Notify Other Candidates of status
(Disposition Candidates)
Negotiate Offer
Accept Offer (Obtain Acceptance)
Complete Background Check
Hire Employee
Send Hire Confirmation Letter (incl. Net ID instr., etc.)
Add Emp to (including Lic + Certs)
- Payroll Sys.
- HR Sys.
Schedule Orientation
Request System Access
Create Dept. Employee File
Request Badge
Complete Internal Notifications as necessary
Notify Manager
Disposition Candidates
Initiate Self-Service Activities
Complete Final Disp. of all Candidates
Pull Advertising
Complete Onboarding
~~Schedule~~ Attend Orientation
Complete Checklist (Emp. v. Manager) (Departmental)
I9
W4
Direct Deposit
Badge
Confirm Employee Completion (Net ID, email, Logon works, ... & Commuter Services

- *Identify*
  Payroll Title & Job Title
- *Develop* Job Description
- *Obtain*
  Funding, Compensation, & Space Approvals
- *Identify*
  Supervising Manager
- *Determine*
  Recruiting Method
- *Assemble*
  Recruiting Resources (e.g., search committee)

Active verbs & nouns

- *For the first time,* the end-to-end process is visible
- A surprise to *everyone* how much work it is, and *how many functions participate!*
- Still no reference to “who or how” – just "*active verb* + noun" (They did a great job!)
- *This is critical to build support for change – it “depersonalises” in a good way!*

For each essential Activity, add "Who," "How," and lots of "Notes"



- Next, add "who" (which role) and "how" (which tool or system function) and "notes."
- Now we have the basics of a to-be process design, and an understanding of which steps will be supported by which system functions – great for understanding if the COTS app will actually work!
- *Jumping into detailed requirements and process flow models would have prevented us from getting here!*

Concept Modelling was really important

Supervisory-Organisational Hierarchy

Position-Based Management

Come to the session to see the simple but effective Concept Models!

Case study: “Why do we hate this application?”

Evaluation

Client conducts feature-based evaluation of system to track job applicants. SuperApp 1.0 is selected.

Configuration

Client spends >10x the purchase price modifying SuperApp to match their business needs.

Realisation

Nobody likes the system very much, including the federal regulator that isn't getting the required info.

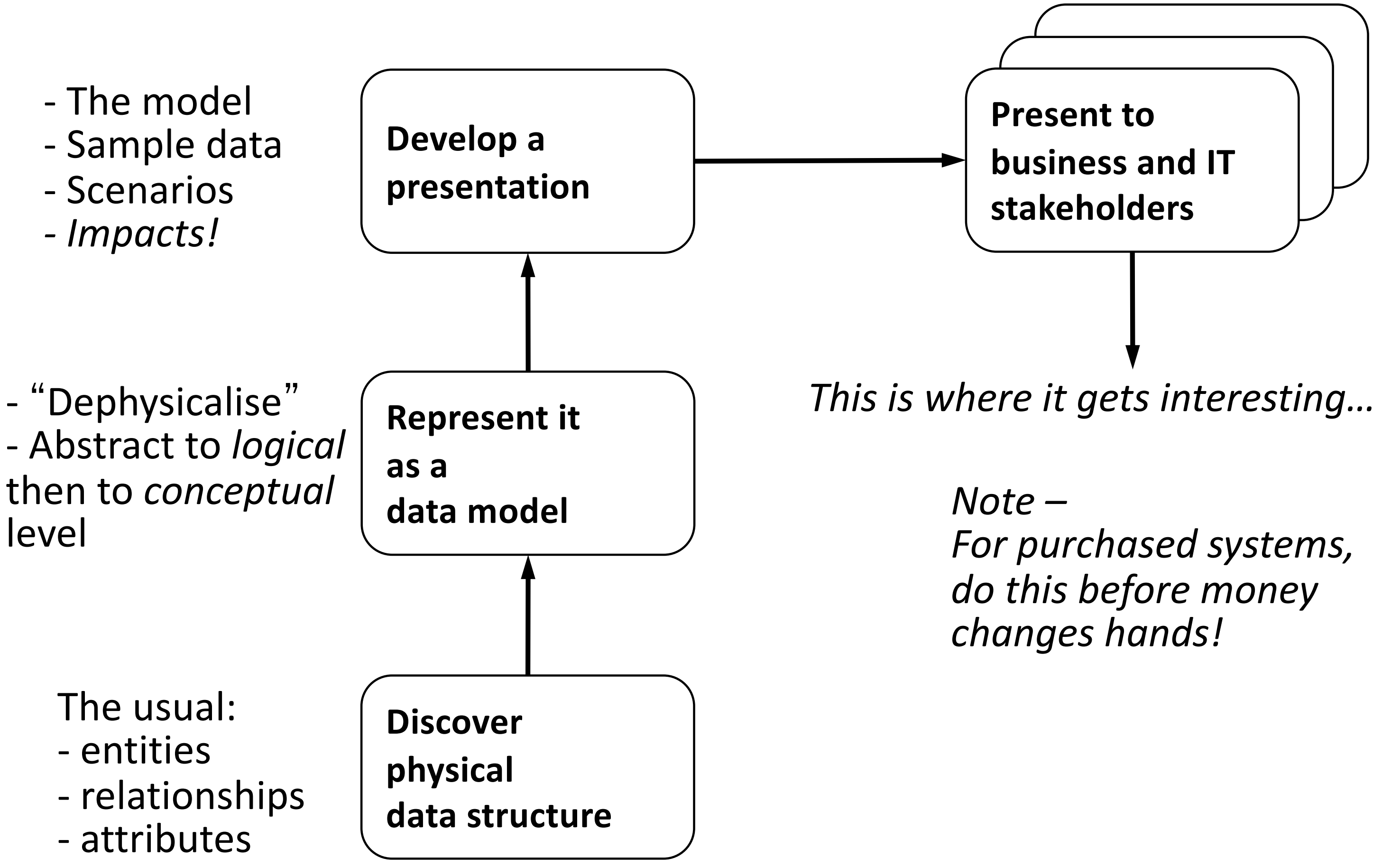
Salvation

Vendor announces SuperApp 2.0, which has a cool new GUI and a host of new features.

Desperation

“If we go to SuperApp 2.0, we'll lose our massive investment in customisation! We need a scapegoat!”

Overview of a typical reverse-engineering effort


- The model
- Sample data
- Scenarios
- *Impacts!*
Develop a presentation
Present to business and IT stakeholders
- "Dephysicalise"
- Abstract to *logical* then to *conceptual* level
Represent it as a data model
This is where it gets interesting…
Note –
For purchased systems, do this before money changes hands!
The usual:
- entities
- relationships
- attributes
Discover physical data structure

Show the business…

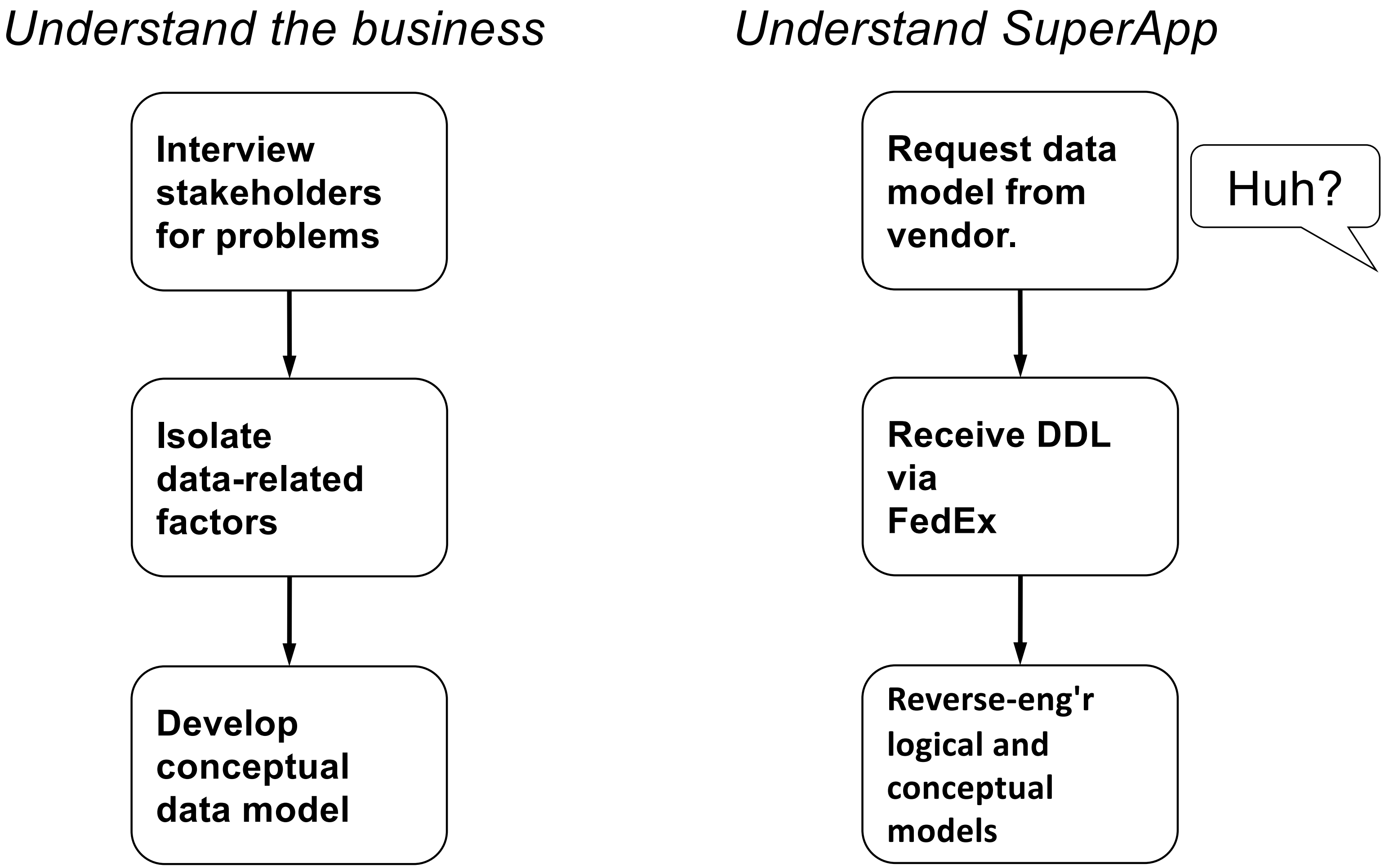
Many possible reactions:
- Horror
- Jubilation
- Glum acceptance, sorrow, shock & awe…

Many possible outcomes
- Adapt - *us*
- Adapt - *it*
- Abandon it
- Retain it

"Oh, nooooo…."

"Gee, that's swell!"

The approach – two parallel streams



Understand the business

Interview stakeholders for problems

Isolate data-related factors

Develop conceptual data model

Understand SuperApp

Request data model from vendor.

Huh?

Receive DDL via FedEx

Reverse-eng'r logical and conceptual models

At the presentation…

How you see the world...

How SuperApp sees the world...

*Drawn,
not projected*

"World view" not "data model"

*Training manager:
"That's the clearest description of what we do I've ever seen. Can we get that for our orientation material?"*

*HR director:
"This has been a revelation!"*

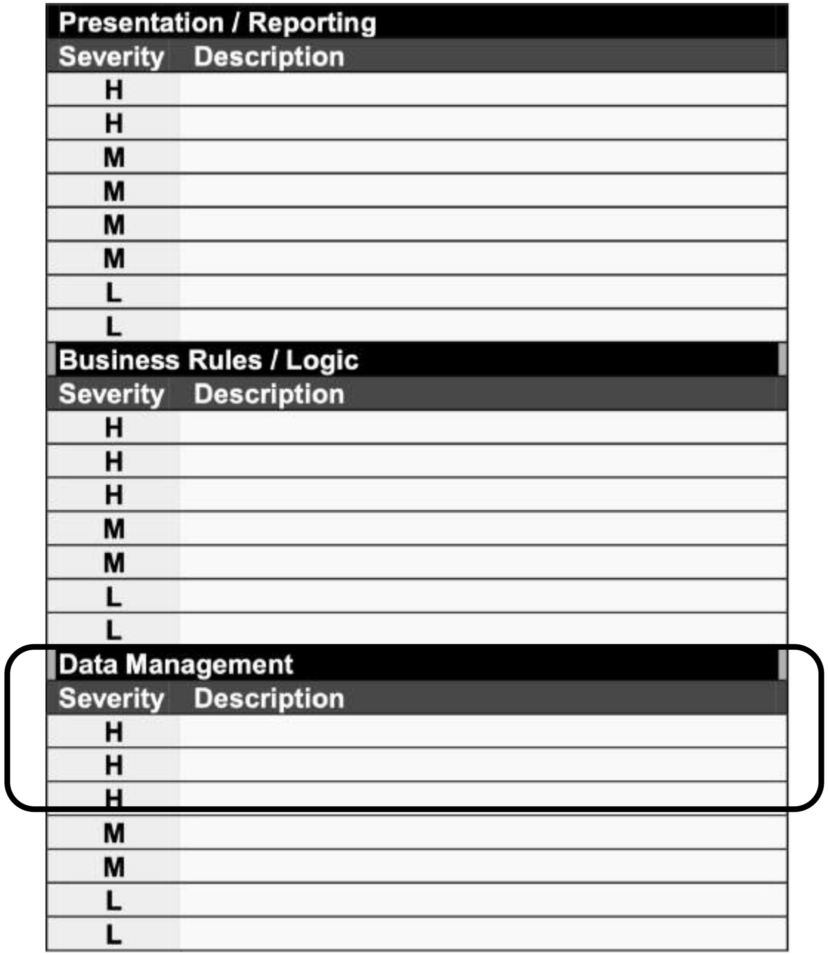
Influencing the vendor

The approach for assessing app and negotiating with the vendor:

- Classify issues into 3 tier framework
- Assign severity
- Focus *only* on “High” severity Data Management issues
- Discuss desired changes with vendor (and have a nice weekend on the East Coast, too!)

If you can't get the data model right, nothing else matters!

Vendor chooses to ignore requests. Company name is now owned by two guys in a garage somewhere overseas.



Presentation / Reporting	
Severity	**Description**
H	
H	
M	
M	
M	
M	
L	
L	
Business Rules / Logic	
Severity	**Description**
H	
H	
H	
M	
M	
L	
L	
Data Management	
Severity	**Description**
H	
H	
H	
M	
M	
L	
L	

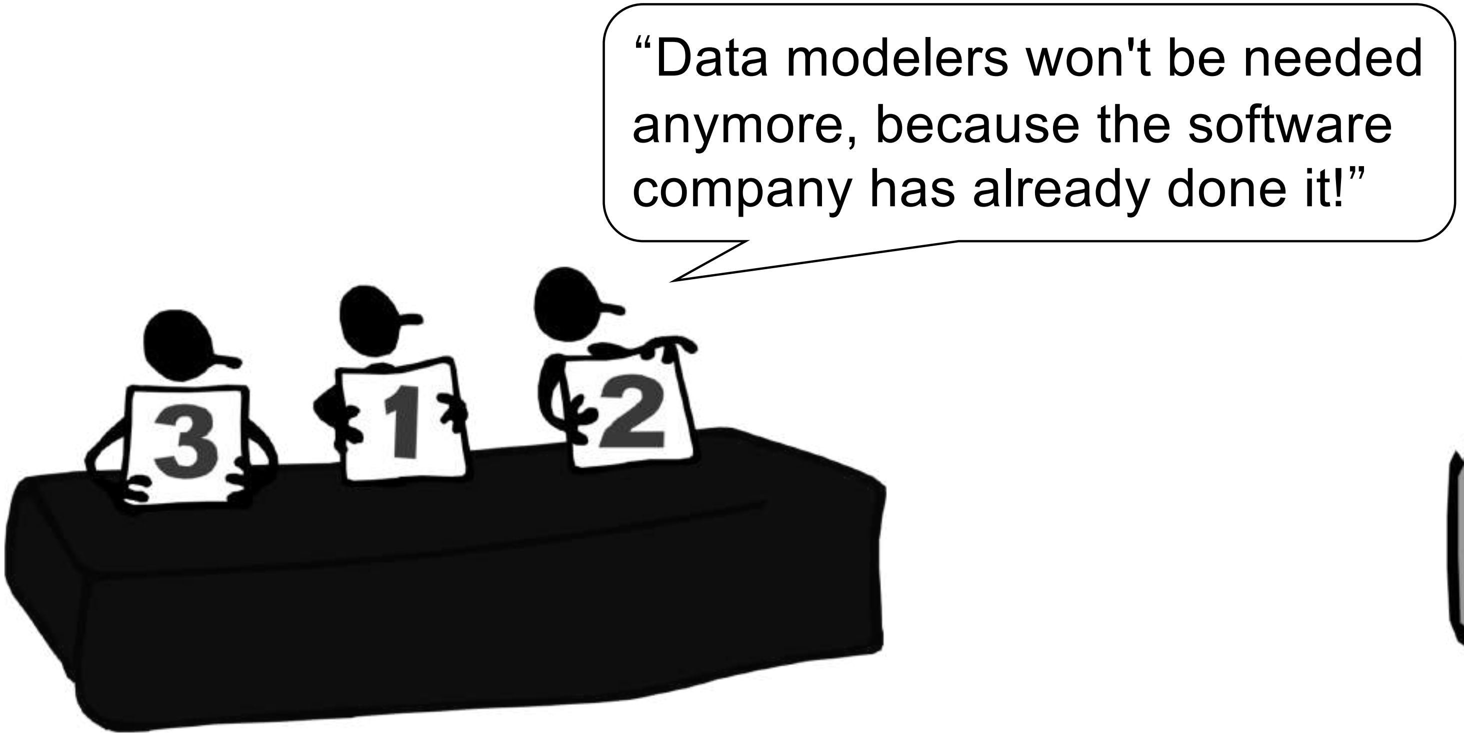
Retrospective

The senior IT manager involved in this project said
“I usually think consultants are a waste of space, but
this was great –
you really delivered value.”

This was a very successful consulting engagement –

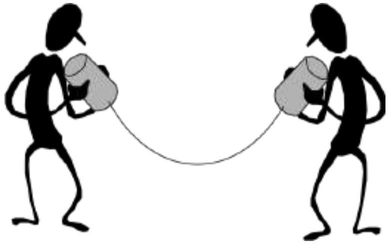
- Think about *why?*
- What did I add they *would not have considered on their own?*

Data modelling – many detractors over the years (but it's where I start!)


“Data modelers won't be needed anymore, because the software company has already done it!”
3
1
2

The beginning of the end?
Various commentators on my
data modelling career, mid-1990s

Redemption!



The client...

Could you come on over and do that thing you do?

That entity data stuff with the boxes and lines

We're implementing something called SAP. Our CEO told us to!

When you did this with our Work Order Management system, we all felt we understood our business better than we ever had.

They say it's a terrible idea, a waste of time, and will you *please stay home!*

Alec...

I guess.
What thing in particular?

Oh, data modelling.
Sure - what's the project?

Ah… sounds familiar.
How can I help?

Nice. And what do your SAP consultants say about me coming out?

I'm on my way.

SAP – using DM for ERP configuration

The problem:

- Application selected by decree
- Desire to understand as-is business to map to package and decide on configuration options
- Client felt they were being coerced by integrator

The approach:

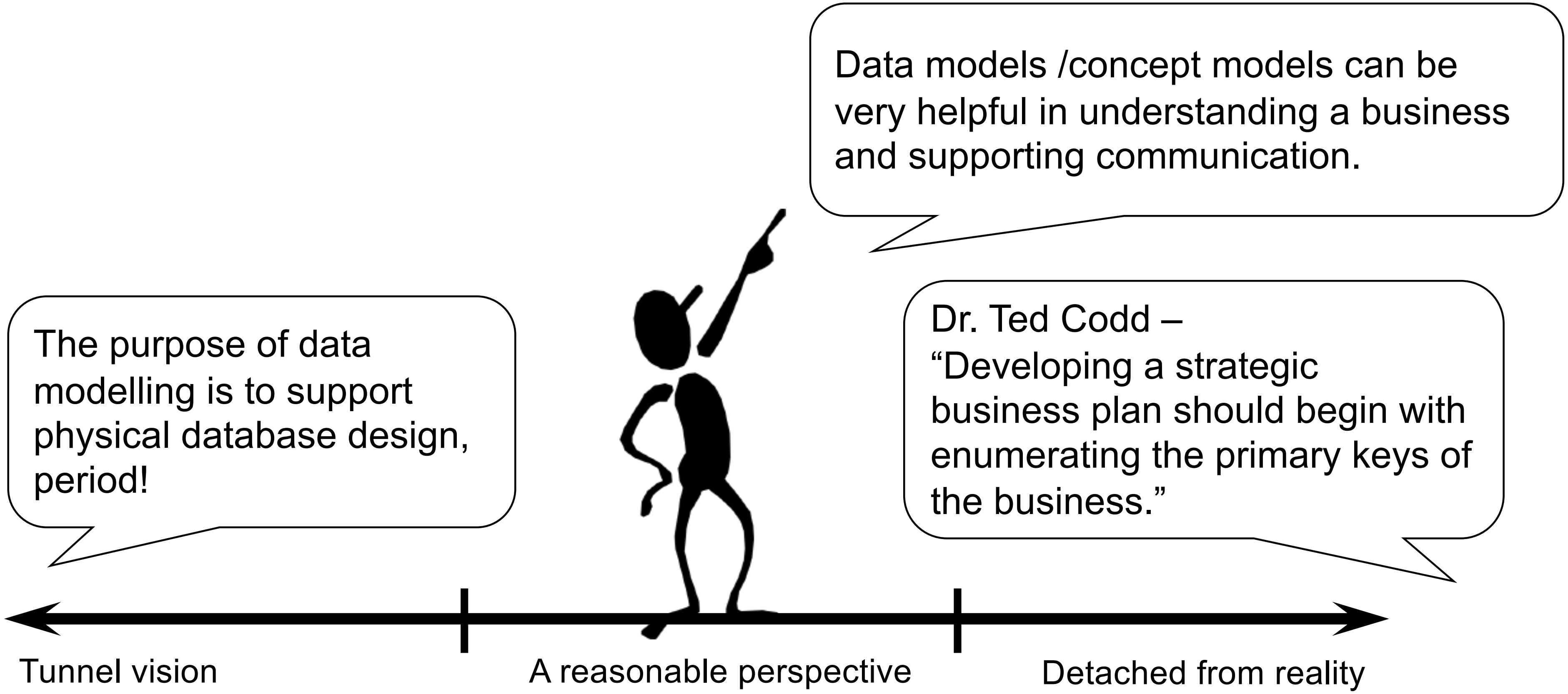
- Team of 7 builds 45 entity *conceptual* data model over two days
- Identify “what's good, what's not good” about current business rules
- Use this knowledge on configuration activities with data model as an overall map

Vendor
Country
Plant
Plant Location
Equipment Item & Type
PO, PO Line Item
Req, Req Line Item
Release, Release Line Item
Work Definition, WD Item
etc. etc. etc.

The key points:

- ***Client-initiated, not IT***
- Now a global showcase account
- Client – “More value from those two days than anything else we did!”
- Me – “I'm not irrelevant!”

A reasonable perspective



Data models /concept models can be very helpful in understanding a business and supporting communication.

The purpose of data modelling is to support physical database design, period!

Dr. Ted Codd –
“Developing a strategic business plan should begin with enumerating the primary keys of the business.”

“Data Model mismatch” –
the most common (*but unrecognised!*) reason for unhappiness with a purchased application

A similar example - purchased application selection

The problem:

- Selection of new Financials app is hopelessly bogged down (and a matrix of almost 1000 “requirements” wasn't helping)
- Worse – *matrix points to the app no one wants!*

The approach:

- Small team builds “thing model” (concept model, ~60 entities total, 15 “core”)
- For each core entity, identify the core life cycle events
- For each event, develop scenario w. data
- Turn over to vendors –*paid* proof of concept
  - “How do you support the data model?”
  - “How do you handle scenarios?”

“Things we track…”
Project, Work Order
Plant, Plant Equipment
Product Type, Product Lot
Product Inventory
Sale, Transfer
Location, Ledger Entity
Financial Category
Responsibility Center
Account, Sub-Account
Fixed Asset

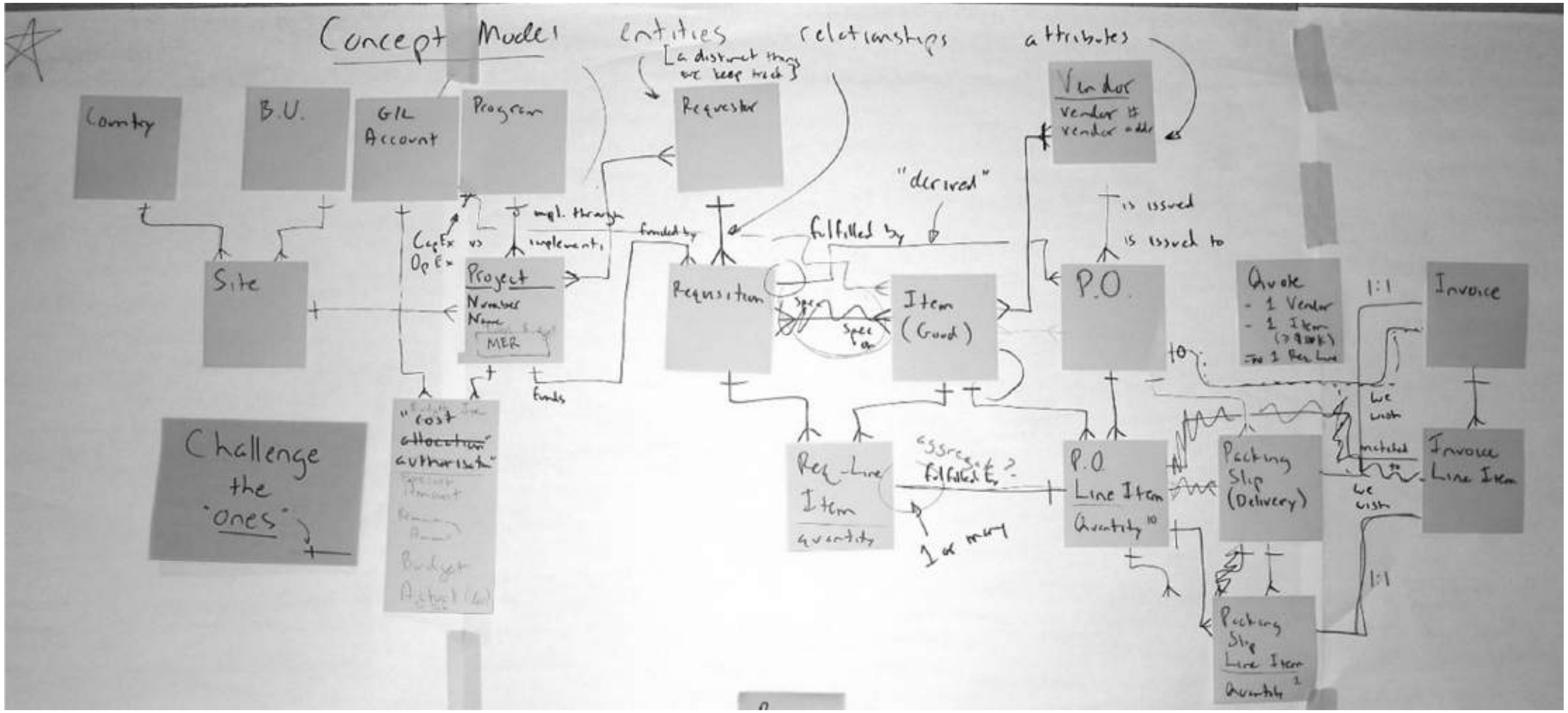
“Events that happen to them…”
Fixed Asset is

- Acquired or Constructed
- Depreciated
- Transferred
- Disposed Of

The key points:

- It worked! – saw how an app would support the business
- Didn't initially call it “data modelling”
- Left vendor some room - “Here's how we'd do it.”

Another example – Concept Model reveals process possibilities



- *Program vs. Project*
- *Site vs. BU Location*
- *Requisition vs. Quote vs. Purchase Order*
- *Invoice/Invoice Line Item vs. Packing Slip/Packing Slip Item*

and many other terms were clarified.

Three incredibly useful patterns

1. Entity definition starting with "anomalies and potential sources of confusion"

2. "Challenge the ones!"

3. The "Types vs. Instances" issue

All will be illustrated at the session in Utrecht –
be there!

Thanks again!

Alec Sharp, West Vancouver, BC, Canada

If you have questions or comments…
don't be shy, get in touch!

- e: asharp@clariteq.com
- t: @alecsharp
- ig: @alecsharp01
- m: +1 604 418-3352