

TECHNICAL DEBT ISN'T TECHNICAL

@EINARWH



WHAT IS TECHNICAL DEBT?

TECHNICAL DEBT AS A MEME

**I SHOULD MAKE
AN INTERNET MEME**

made on imgur

MEMETICS

SELF-REPLICATING UNIT OF THOUGHT

**COPIED FROM
BRAIN TO BRAIN**

VIRUS OF THE MIND

WHAT MAKES A MEME **SUCCESSFUL?**

SURVIVAL OF THE FITTEST?

GREATEST
EXPLANATION
POWER?

MOST EFFECTIVE AT
REDUCING COGNITIVE
DISSONANCE?

THE BEST IDEAS
VS
THE IDEAS THAT
MAKE US FEEL BEST

WHAT IS TECHNICAL DEBT?



SHIPPING FIRST TIME CODE IS LIKE GOING INTO DEBT.
A LITTLE DEBT SPEEDS DEVELOPMENT SO LONG AS IT
IS PAID BACK PROMPTLY WITH A REWRITE... THE
DANGER OCCURS WHEN THE DEBT IS NOT REPAYED.
EVERY MINUTE SPENT ON NOT-QUITE-RIGHT CODE
COUNTS AS INTEREST ON THAT DEBT. ENTIRE
ENGINEERING ORGANIZATIONS CAN BE BROUGHT TO A
STAND-STILL UNDER THE DEBT LOAD OF AN
UNCONSOLIDATED IMPLEMENTATION.

WHAT MAKES TECHNICAL DEBT A GOOD METAPHOR?

**GIVES US LANGUAGE
TO TALK ABOUT
SOMETHING IMPORTANT!**

PUTS **QUALITY** ON THE
AGENDA!

**MAKES THE INVISIBLE
VISIBLE!**

**THANK YOU
WARD CUNNINGHAM!**

WHAT MAKES TECHNICAL DEBT A PROBLEMATIC METAPHOR?

**MONETARY DEBT
IS QUANTIFIED**

HOW MEASURABLE IS TECHNICAL DEBT?

HOW **BIG** IS OUR TECHNICAL DEBT?

ARE WE
TECHNICALLY SOLVENT?

CAN WE GO
TECHNICALLY BANKRUPT???

WHAT IS TECHNICAL DEBT REALLY?

WHAT DOES TECHNICAL DEBT LOOK LIKE?

"UGLY CODE"?

WHAT CAUSES TECHNICAL DEBT?

WHAT MAKES THE CODE UGLY?

LACK OF TIME?

MY PROJECT MANAGER DOESN'T UNDERSTAND ME?

THE UX PEOPLE ARE UNREALISTIC DREAMERS?

**THE PREVIOUS DEVELOPER
WAS INCOMPETENT?**

WHAT DOES TECHNICAL DEBT LOOK LIKE?

HEAPS OF FLAGS, BRANCHES AND INDIRECTIONS

LOOP THE LUU

TRAP YOURSELVES

I'LL JUST ADD
ANOTHER IF BRANCH

**DIFFICULT TO DETERMINE
WHERE ONE THING ENDS
AND ANOTHER BEGINS**

**DIFFICULT TO ISOLATE
CODE RELATED TO A
SINGLE FEATURE**

**DIFFICULT TO PREDICT
THE EFFECT OF CHANGES**

DIFFICULT TO REASON ABOUT THE CODE

**WHAT DOES THIS
INDICATE?**

**THAT THE PRODUCT OWNER
DOESN'T UNDERSTAND
TECHNOLOGY?**

THAT THE PREVIOUS
DEVELOPER USED THE
WRONG FRAMEWORK FOR
DEPENDENCY INJECTION?

WHY
DO YOU
THINK
THAT WE SHOULD
INTRODUCE MICRO
SERVICES?

WHY
DO WE NEED
A NOSQL DATABASE?

THAT WE SHOULD GO
SERVERLESS?

MOVE EVERYTHING TO
KUBERNETES?

NO.

(THAT MUCH WAS OBVIOUS.)

BUT IT'S ALSO OBVIOUS THAT
SOMETHING HAS GONE
FUNDAMENTALLY WRONG

**SOMETHING HAS GONE
WRONG IN THE PROCESS OF
CAPTURING THE BUSINESS
DOMAIN IN SOFTWARE**

**SOMETHING HAS GONE
WRONG IN THE PROCESS OF
UNDERSTANDING THE
BUSINESS DOMAIN**

**SOMETHING HAS GONE
WRONG IN THE PROCESS OF
MAKING AND ENFORCING
DECISIONS**

**SOMETHING HAS GONE
WRONG IN THE PROCESS OF
FINDING A SUITABLE
REPRESENTATION OF THE
DOMAIN IN SOFTWARE**

**WE HAVE A WORD
FOR THIS PROCESS**

MODELLING

MODELLING HAS FAILED

**WHAT IS THE NAME OF THE
TOOL WE USE TO **CONTAIN**
AND **CONQUER** COMPLEXITY?**

ABSTRACTION

**WHAT IS THE NAME OF THE
TOOL WE USE TO **CREATE** AND
COMPOUND COMPLEXITY?**

ABSTRACTION

**MODELLING IS APPLYING
ABSTRACTION TO A
PROBLEM DOMAIN**

**MODELLING IS DEFINING
WHAT THE PROBLEM
DOMAIN IS**

WORKING CONTINUOUSLY TO
UNDERSTAND AND ARTICULATE
THE PROBLEM TO SOLVE IS NOT
BIG DESIGN UP FRONT

**“YOU'RE WISE TO MAKE THE SOFTWARE REFLECT YOUR
UNDERSTANDING AS BEST YOU CAN”**

HOW CAN WE PAY OFF TECHNICAL DEBT?

(EVERYBODY NOW)

REFACTORING!

WHICH REFACTORING FIXES A TANGLED MESS OF WEAK ABSTRACTIONS?





Using the Catalog ►

Tags

- associations
- encapsulation
- generic types
- interfaces
- class extraction
- GOF Patterns
- local variables
- vendor libraries
- errors
- type codes
- method calls
- organizing data
- inheritance
- conditionals
- moving features
- composing methods
- defining methods

Books

- Refactoring
- Ruby Edition
- appear only [explain](#)

#

- [Add Parameter](#)
- [Change Bidirectional Association to Unidirectional](#)
- [Change Reference to Value](#)
- [Change Unidirectional Association to Bidirectional](#)
- [Change Value to Reference](#)
- [Collapse Hierarchy](#)
- [Consolidate Conditional Expression](#)
- [Consolidate Duplicate Conditional Fragments](#)
- [Decompose Conditional](#)
- [Duplicate Observed Data](#)
- [Dynamic Method Definition](#)
- [Eagerly Initialized Attribute](#)
- [Encapsulate Collection](#)
- [Encapsulate Downcast](#)
- [Encapsulate Field](#)
- [Extract Class](#)
- [Extract Interface](#)
- [Extract Method](#)
- [Pull Up Constructor Body](#)
- [Pull Up Field](#)
- [Pull Up Method](#)
- [Push Down Field](#)
- [Push Down Method](#)
- [Recompose Conditional](#)
- [Remove Assignments to Parameters](#)
- [Remove Control Flag](#)
- [Remove Middle Man](#)
- [Remove Named Parameter](#)
- [Remove Parameter](#)
- [Remove Setting Method](#)
- [Remove Unused Default Parameter](#)
- [Rename Method](#)
- [Replace Abstract Superclass with Module](#)
- [Replace Array with Object](#)
- [Replace Conditional with Polymorphism](#)
- [Replace Constructor with Factor](#)

A REWRITE WILL END UP WITH THE SAME
PROBLEMS AS THE ORIGINAL UNLESS
YOU CLOSE THE UNDERSTANDING GAP.

— @SARAHMEI

**NO REFACTORING
WITHOUT REMODELLING**

**ANYTHING ELSE IS
DEVELOPER HUBRIS**

SOFTWARE CRAFTSMAN TO THE RESCUE





**SO-CALLED BEST PRACTICES
ARE NOT ENOUGH!**

**CLEAN CODE CAN NOT
AND WILL NOT SAVE A
ROTTEN MODEL!**

**YOU WILL WASTE
TIME AND MONEY!**

**FIX THE MODEL FIRST
THEN THE CODE**

ARE WE DONE NOW?

HOW DO WE FIX THE MODEL?

WHAT CAUSES MODELLING DEBT?

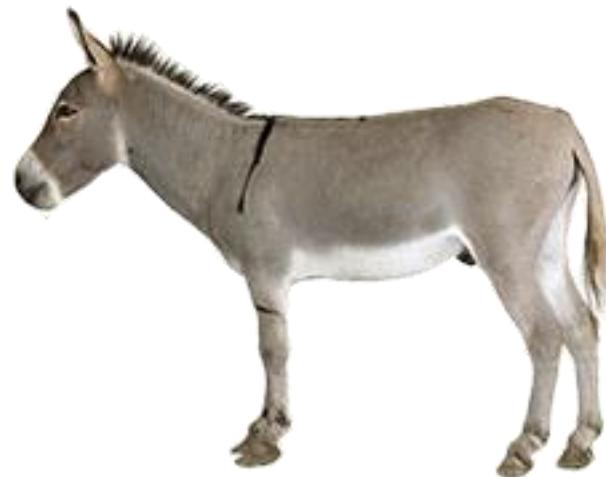
(OMG HERE WE GO AGAIN!)

THE STORY OF EQUEST

A HORSE



A NEW HORSE



A SHORT, STRONG AND STUBBORN HORSE

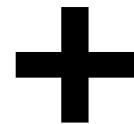


A HORSE?

A HORSE?



A HORSE?



A REGULAR HORSE



A TYPICAL HORSE



A TRUE HORSE



AN OLD-FASHIONED HORSE



A HORSE-HORSE



```
if (horse.IsShort && horse.IsStubborn)
{
    // Logic for the new horse case.
}
else
{
    // Regular horse code here.
}
```

BUGS!

```
if (horse.IsShort &&
    horse.IsStubborn &&
    horse.Sound == Sound.HeeHaw)
{
    // Logic for the new horse case.
}
else
{
    // Regular horse code here.
}
```

A NEW NEW HORSE



**THE KIND OF HORSE THAT
IS THE OFFSPRING OF A
REGULAR HORSE-HORSE**



**AND A SHORT AND STUBBORN
HORSE THAT GOES HEE-HAW**

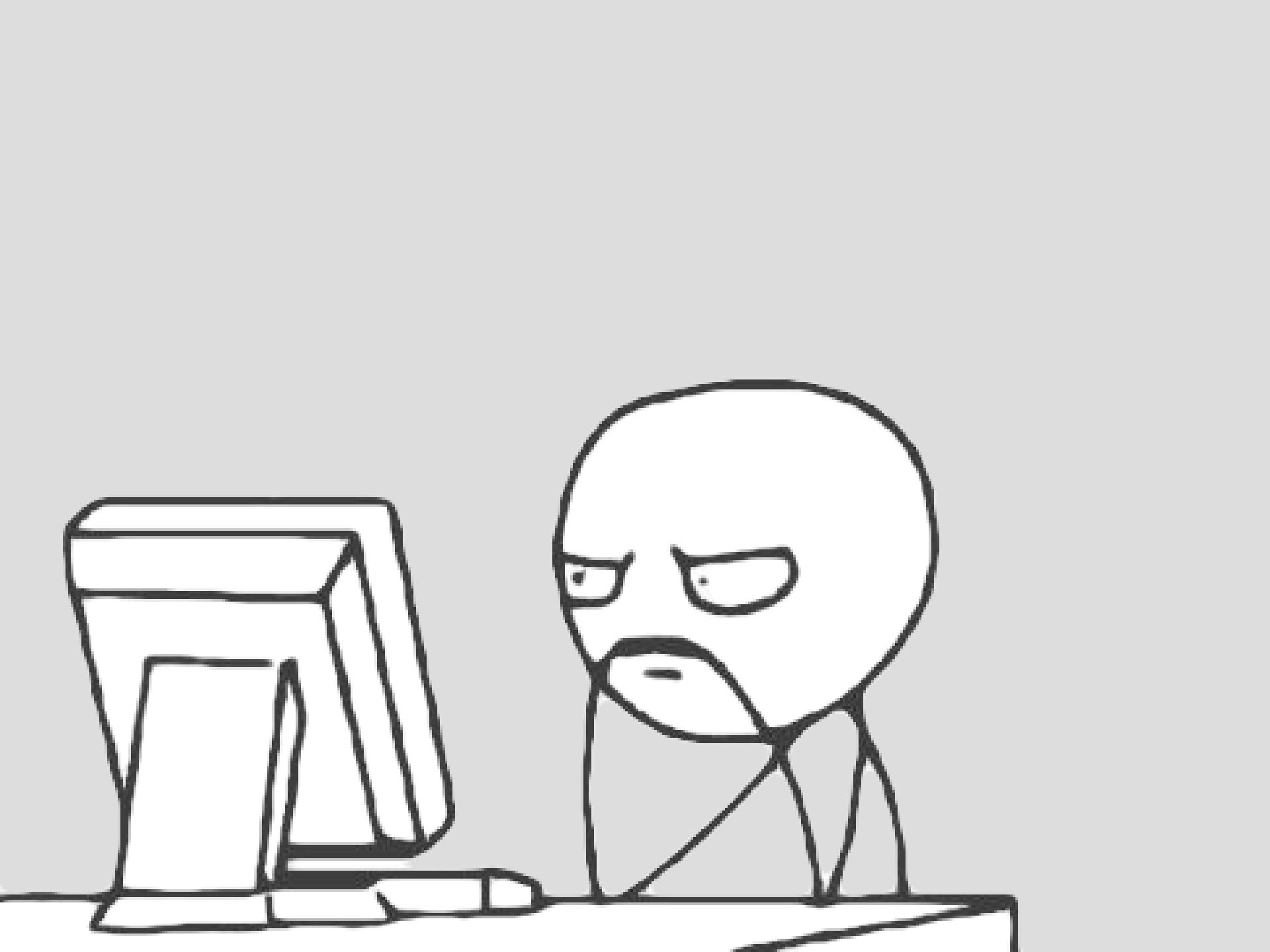
```
if (horse.IsShort &&
    horse.IsStubborn &&
    horse.Sound == Sound.HeeHaw) ||
(horse.Sire.IsShort &&
 horse.Sire.IsStubborn &&
 horse.Sire.Sound == Sound.HeeHaw) ||
(horse.Dam.IsShort &&
 horse.Dam.IsStubborn &&
 horse.Dam.Sound == Sound.HeeHaw))
{
    // Logic for both the new horse
    // and the new-new horse!
} else
{
    // Really regular horse code here.
}
```

BUGS!

**IS IT THE MOTHER
OR THE FATHER**

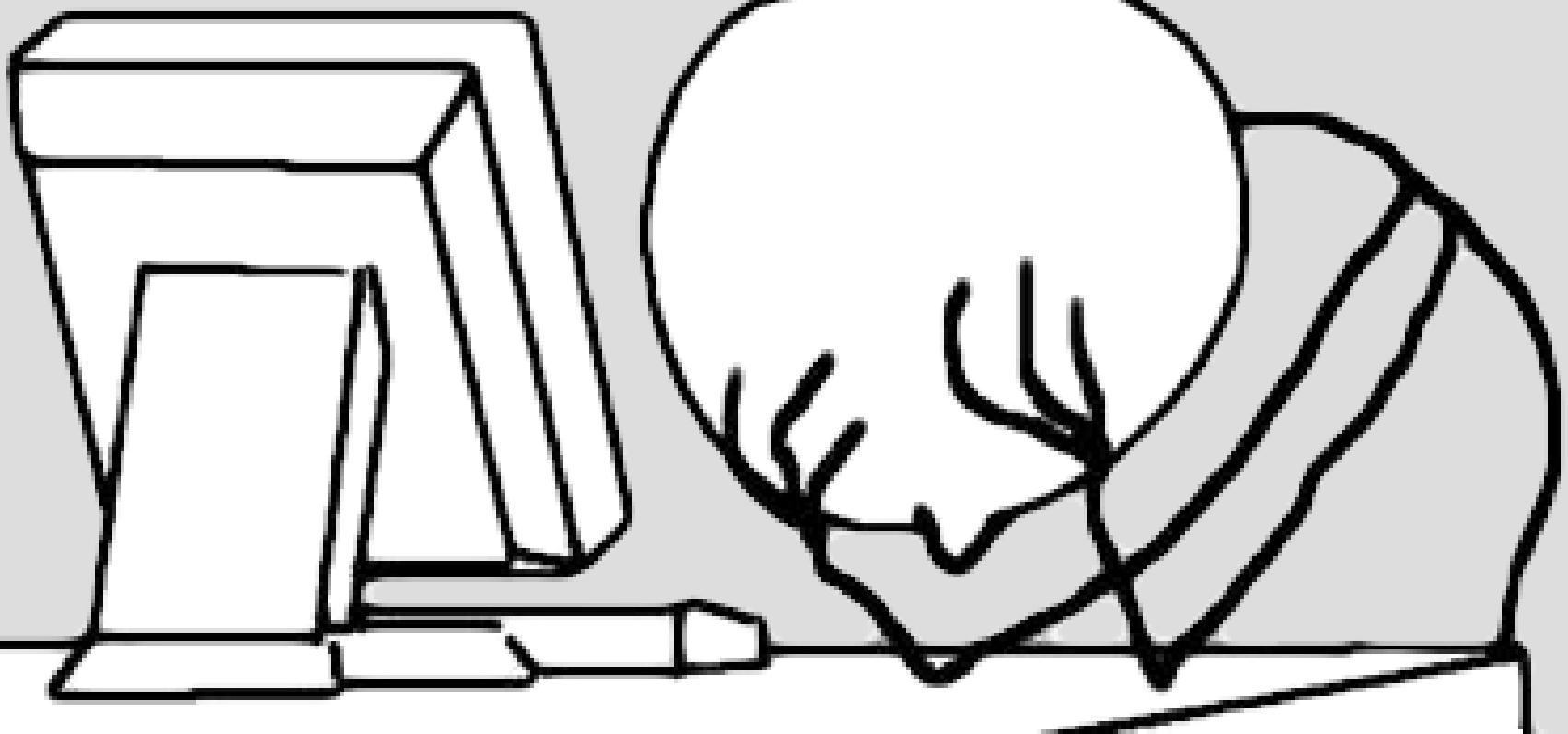


**THAT IS STUBBORN AND
STRONG AND GOES HEE-HAW?**



A NEW NEW NEW HORSE





RECOMMENDED READING
**WHAT, IF ANYTHING,
IS A ZEBRA?**
SJ GOULD

GIBT ES LEPORIDEN?



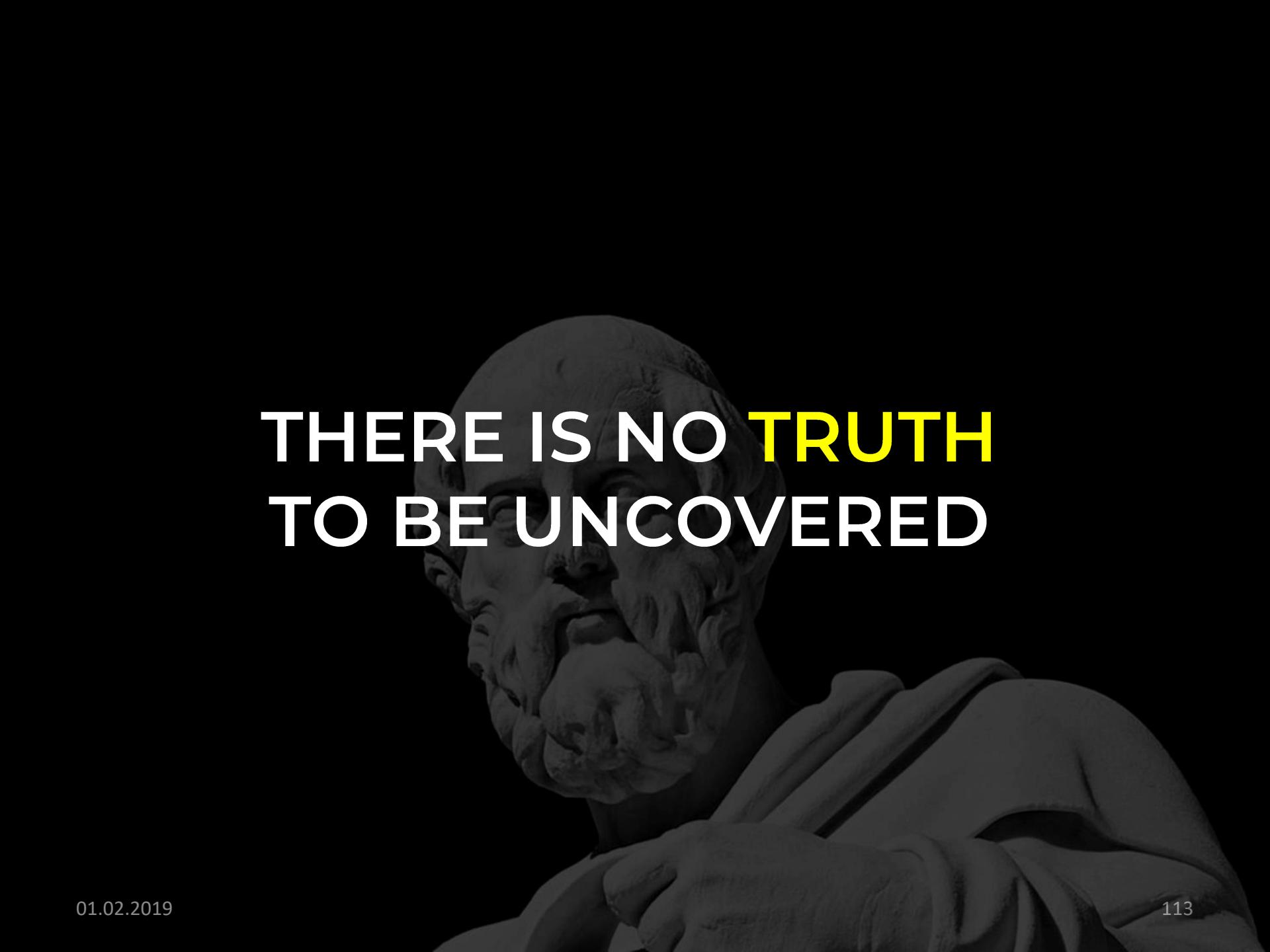
WHAT, IF ANYTHING, IS A RABBIT?



WHAT, IF ANYTHING, IS A ZEBRA?



STRIPES DO NOT A ZEBRA MAKE



**THERE IS NO TRUTH
TO BE UNCOVERED**

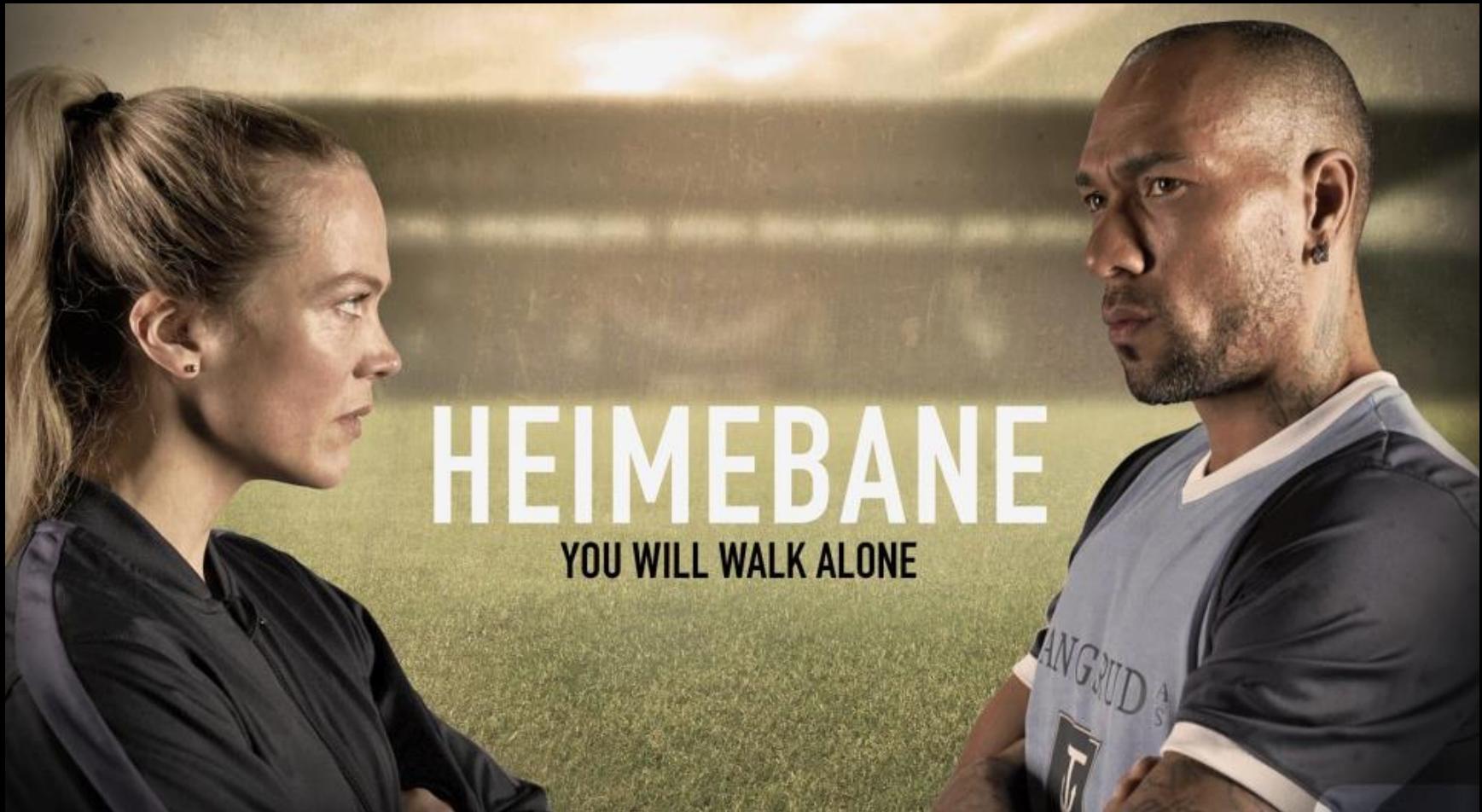
THERE ARE MORE OR LESS
USEFUL MODELS

**MODELLING IS AS MUCH ABOUT
DEFINING AS DISCOVERING**

THE STORY OF NRK TV

**WHAT, IF ANYTHING,
IS A TV SHOW?**

A TV SHOW



ANOTHER TV SHOW



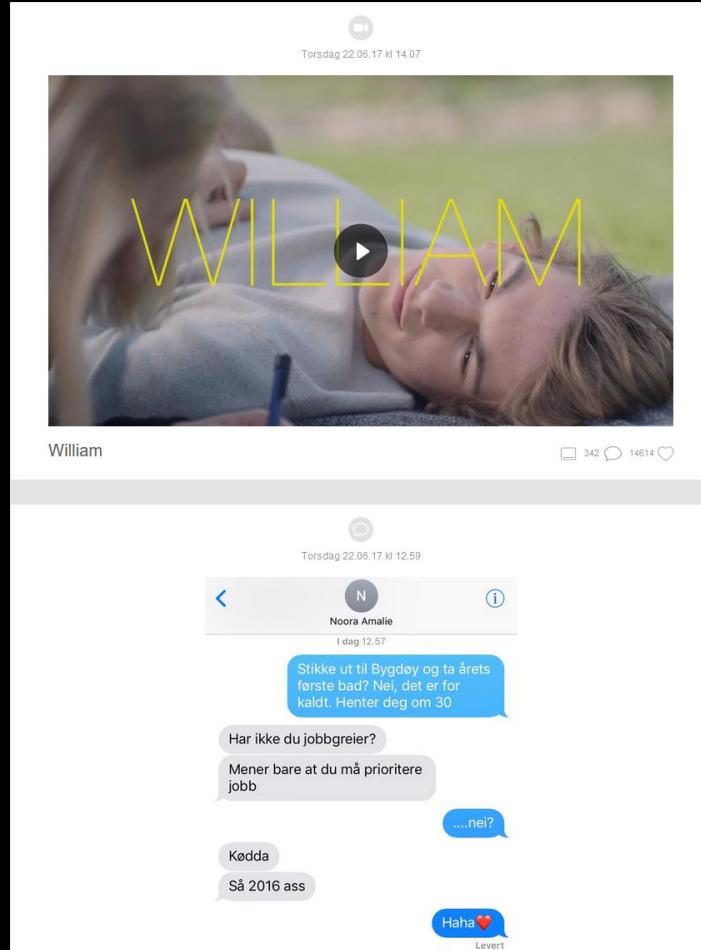
YET ANOTHER TV SHOW



AND YET ANOTHER TV SHOW



A TV SHOW?



HOW MANY KINDS OF TV SHOWS ARE THERE?



WHICH CONCEPTS SHOULD WE USE?

**HOW CAN WE ENABLE
OURSELVES TO TALK
PRECISELY?**

**WHAT DO WE WANT
TO EXPRESS?**

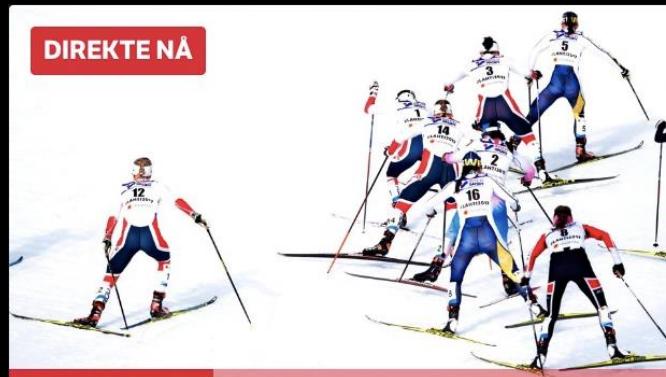
LIVE EVENTS

14:45



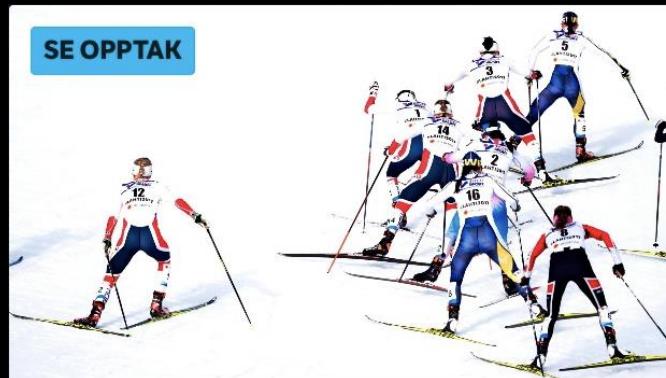
PLANNED

15:30



LIVE

17:00



AFTER

LIVE BROADCAST



20.25

Norge nå

20.55 Distriktsnyheter

21.00 Dagsrevyen 21

21.25 Debatten ...

22.05 Innafor: Dommedag

22.55 Distriktsnyheter

18:52:57

20:24:59

21:52:18



hjelp



DIREKTE

SO TO SUMMARIZE...

**YOU CAN WATCH
LIVE ON DEMAND**

**YOU CAN WATCH
ON DEMAND LIVE**

YOU CAN WATCH
LIVE LIVE

LANGUAGE IS TERRIBLE!

MODELLING IS HARD!

MODELLING IS HARD FOR TWO REASONS

#1

THE STUFF THAT IS MODELLED

#2

THOSE DOING THE
MODELLING

WHY MODELLING IS HARD

#1

THE STUFF THAT IS MODELLED

**REALITY HAS NO
INTEREST IN BEING
MODELLED**

**REALITY IS NOT TIDY
AND ORGANIZED**

**REALITY IS NOT EVEN
CONSISTENT**

**REALITY IS MESSY AND
ORGANIC, SLIPS AWAY
AND EVOLVES**

WE ARE TRYING TO IMPOSE
STRUCTURE ON SOMETHING
UNSTRUCTURED

**INNOVATION MEANS
BREAKING MODELS**

RECOMMENDED READING

DATA AND REALITY

WILLIAM KENT

IDENTITY

(WHAT IS ONE THING?)

EQUALITY

(WHEN ARE TWO THINGS EQUAL?)

(WHEN ARE THEY THE SAME THING?)

ONE THING?



OR TWO?

CHANGE

(WHEN DO TWO THINGS BECOME DISTINCT?)

(WHEN HAS ONE THING BECOME ANOTHER?)

CATEGORY

[WHICH CATEGORY DOES A THING BELONG TO?]

[CAN A THING CHANGE CATEGORY?]

[IS IT THE SAME THING IF IT CHANGES CATEGORY?]

RECOMMENDED READING

**WOMEN, FIRE AND
DANGEROUS THINGS**

GEORGE LAKOFF

THE CLASSICAL THEORY
OF CATEGORIZATION IS
BASED ON PROPERTIES

CLASSES IN PROGRAMMING LANGUAGES LIKE **SIMULA AND C#**

NECESSARY AND
SUFFICIENT PROPERTIES
DETERMINE CATEGORY
MEMBERSHIP

**WHAT DOES A BIRD
LOOK LIKE?**







SOME MEMBERS OF A
CATEGORY ARE MORE
TYPICAL THAN OTHERS

A SPARROW IS MORE
BIRDY THAN A PENGUIN

A SPARROW IS MORE
BIRDY THAN A SHOEBILL

A SPARROW IS PROBABLY
THE BIRDIEST BIRD THERE IS

**PROPERTY-BASED
CLASSIFICATION IS
LIMITED**

STRIPES DO NOT A ZEBRA MAKE

ELEANOR ROSCH PROTOTYPE THEORY

HOW UNTYPICAL IS TOO UNTYPICAL?

HOW UNEQUAL IS TOO UNEQUAL?

WHEN DOES A DONKEY BECOME A DONKEY?

IT DEPENDS!

WHO'S ASKING?

WHO'S ANSWERING?

THE DONKEY DOESN'T CARE!

A DONKEY BY ANY OTHER
NAME WOULD SMELL
THE SAME

WE PUT THINGS IN
CATEGORIES TO **MAKE**
SENSE OF REALITY

GENERALIZATION IS
A HYPOTHESIS
ABOUT THE FUTURE

**BEWARE ARTIFICIAL
AD-HOC
GENERALIZATIONS!**

I DO NOT BELIEVE THAT NATURE FRUSTRATES US
BY DESIGN, BUT I REJOICE IN HER INTRANSIGENCE
NONETHELESS.

— SJ GOULD

WHY MODELLING IS HARD

#2

THOSE DOING THE MODELLING

WHO MAKES SOFTWARE?

WHO OWNS TECHNICAL DEBT?

THE REST OF THE
ORGANIZATION WOULD
LOVE FOR **TECHNICAL DEBT**
TO BE THE PROBLEM OF THE
IT DEPARTMENT

**IF YOU REMAKE AWFUL SOFTWARE FROM SCRATCH
WITHOUT CHANGING THE CULTURE THAT CREATED
IT YOU'LL REMAKE AWFUL SOFTWARE.**

— **@MALK_ZAMETH**

RECOMMENDED READING
HOW DO COMMITTEES INVENT?
MEL CONWAY

CONWAY'S HOMOMORPHISM

AN ORGANIZATION
TENDS TO PRODUCE
SELF-PORTRAITS IN THE
SYSTEMS IT CREATES

INVERSE CONWAY MANEUVER REORGANIZE TO ENABLE ARCHITECTURAL CHANGE

**SYSTEMS ARE
ORGANIZATIONAL
CONCRETE**

COROLLARY TO CONWAY'S LAW:
IF YOU DESIGN A SYSTEM, BUT YOU DIDN'T DESIGN
THE ORGANIZATION STRUCTURE, YOU'RE NOT THE
SYSTEM'S DESIGNER.

— **@MATHIASVERRAES**

THE DIVIDE BETWEEN
BUSINESS AND IT IS
PROBLEMATIC

**SYSTEM ARCHITECTURE IS
ENTERPRISE ARCHITECTURE
AND VICE VERSA**

**WE HAVE COME A LONG
WAY FROM UGLY CODE!**

WHY DO WE STILL TALK ABOUT TECHNICAL DEBT?

THE CONSEQUENCES ARE
MADE **MANIFEST** AND **VISIBLE**
AT THE TECHNICAL LEVEL

**DEVELOPERS LOOK FOR
TECHNICAL SOLUTIONS TO
SOFT PROBLEMS**

THE SOFT PROBLEMS ARE
THE HARD PROBLEMS

**TECHNICAL TASKS
ARE TEMPTING!**

**THEY FIT OUR TRAINING
AND EXPERTISE**

THEY INVOLVE LITTLE IN
TERMS OF PSYCHOLOGY
AND POLITICS

THEY RARELY LEAD
TO CONFLICT

**TECHNICAL WORK IS
PUZZLE SOLVING**

**MODELLING WORK
REVEALS TENSIONS**

YOU MAY HAVE TO
REORGANIZE IN ORDER TO
BE ABLE TO **REFACTOR**

WHAT MAKES A MEME SUCCESSFUL?

GREATEST EXPLANATION POWER?

MOST EFFECTIVE AT
REDUCING COGNITIVE
DISSONANCE?

THERE ARE
NO TECHNICAL SOLUTIONS
TO TECHNICAL DEBT

TECHNICAL DEBT
ISN'T TECHNICAL