Problems during system integration: When architecture and contracts meet

ALEJANDRO SALADO Kayser-Threde GmbH

Disclaimer:

The views and opinions expressed during this presentation are those of the presenter and do not necessarily reflect the official position of Kayser-Threde GmbH or any company of the OHB group.

Who IS called Peter?

Who IS NOT called Peter?

Who would ACCEPT a NOT validated unit?

Who would NOT ACCEPT a NOT validated unit?

Who PROVIDES requirements to the supplier?

Who does NOT PROVIDE requirements to the supplier?

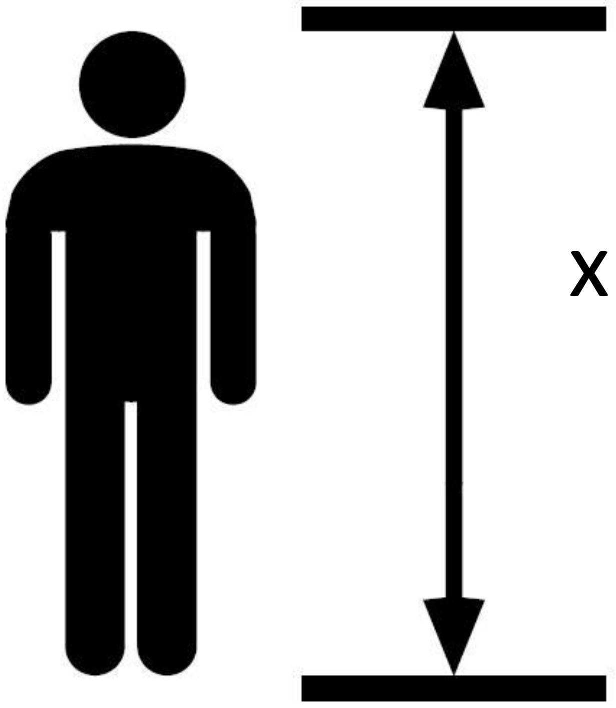
PRINCIPLES

The audience shall congratulate the presenter.

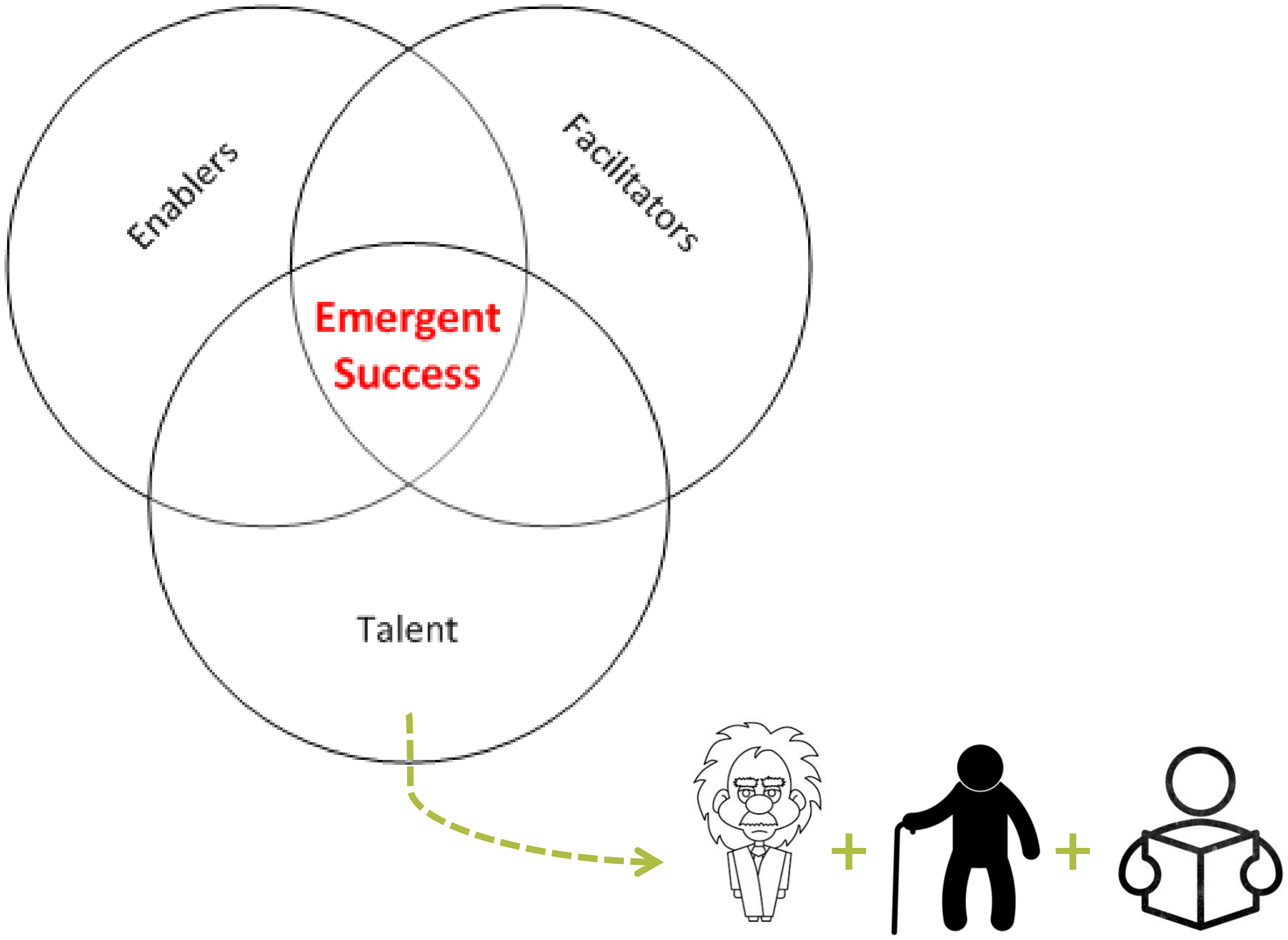
This is a requirement!

Output = 2 x Input A + Input B

This is a requirement!



This is a requirement!


Enablers
Facilitators
Emergent Success
Talent
+
+

The world is UNCERTAIN

FAILURE is a non-preferred **POSSIBILITY**

MEAT

VERIFICATION

Are we building it right?

VALIDATION

Are we building the right thing?

Is that all...?!

V&V activities

Does this test support…?

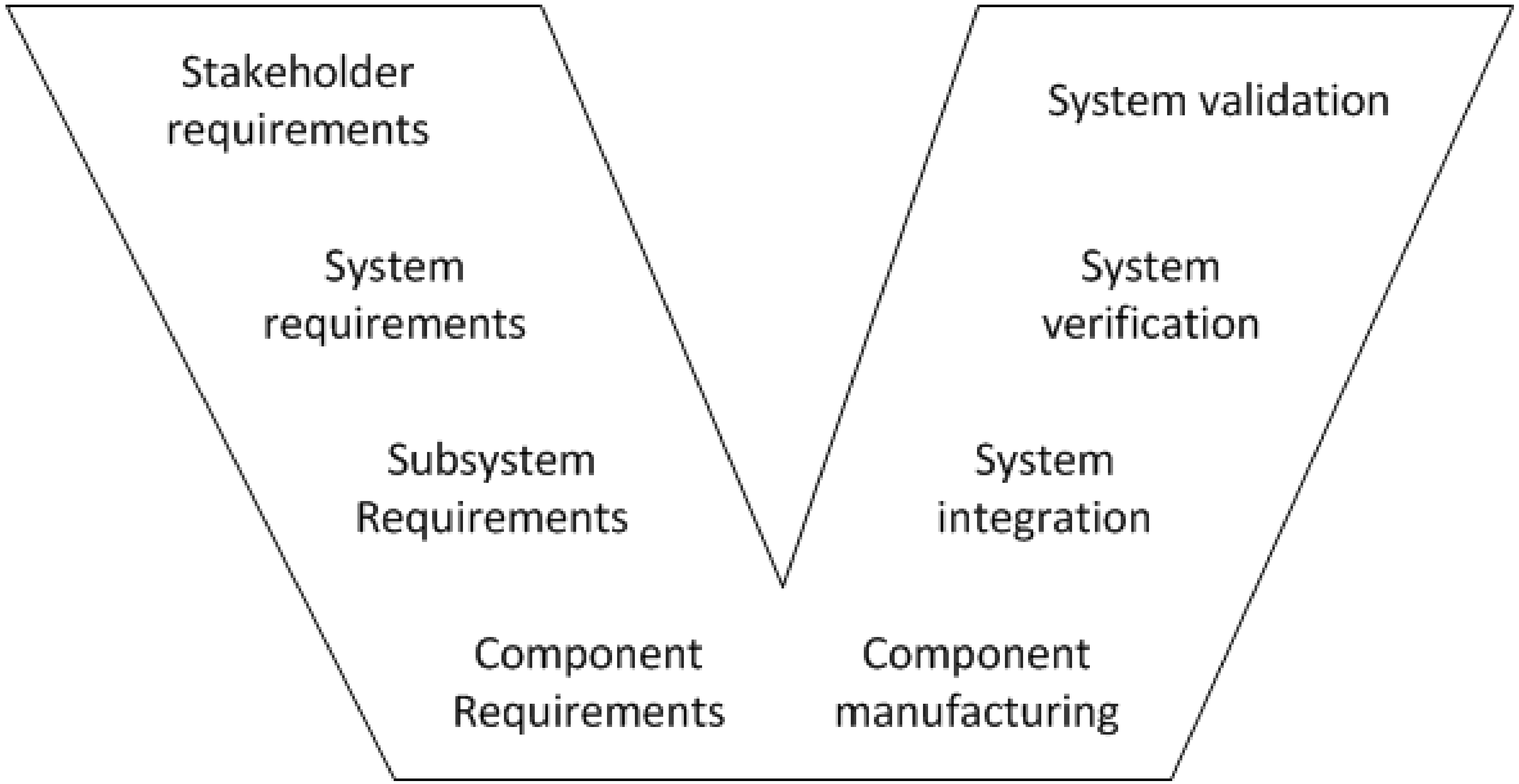
a) Validation
b) Verification
c) Both
d) None

V&V engineer

You verify the design spec and validate the customer spec…

V&V plan

Fuzzyness



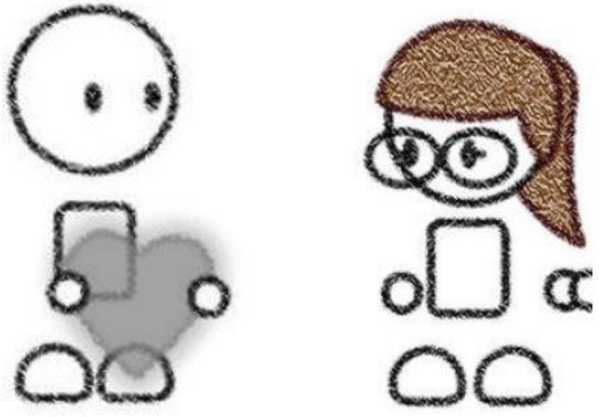
Hierarchical?

What is system integration?

A boy sees a girl...

...or vice versa,

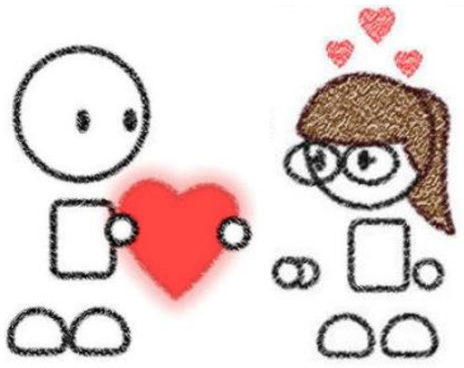
... or any combination.



Begin evaluation...

Size of X is between Y and Z.
Color of A is B.
Hobbies include K, J, and L.
Interests exclude E.
Social status includes T and U, excludes H.

This girl is great!



☑ Requirements are VERIFIED!

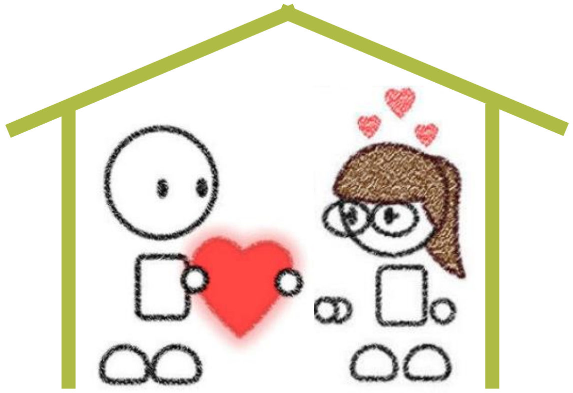
Your mother... says she is not good enough for you.

Your friends... are jellous, the girl is awesome.

Your ex-girlfriend... stops talking to you.

Is she THE girl?

How do you know it? What do you do?



You (both) INTEGRATE!

$\updownarrow$

She is **THE** girl $\longrightarrow$ The girl is **validated**!

☑ Requirements were correct!

$$SI_n \leftrightarrow \bigcup Validation_{Component_{n-1}}$$

… come on, it is only one formula and we are engineers!

REAL **LIFE**

Details and details and details and more details...

We need to release the specification now.

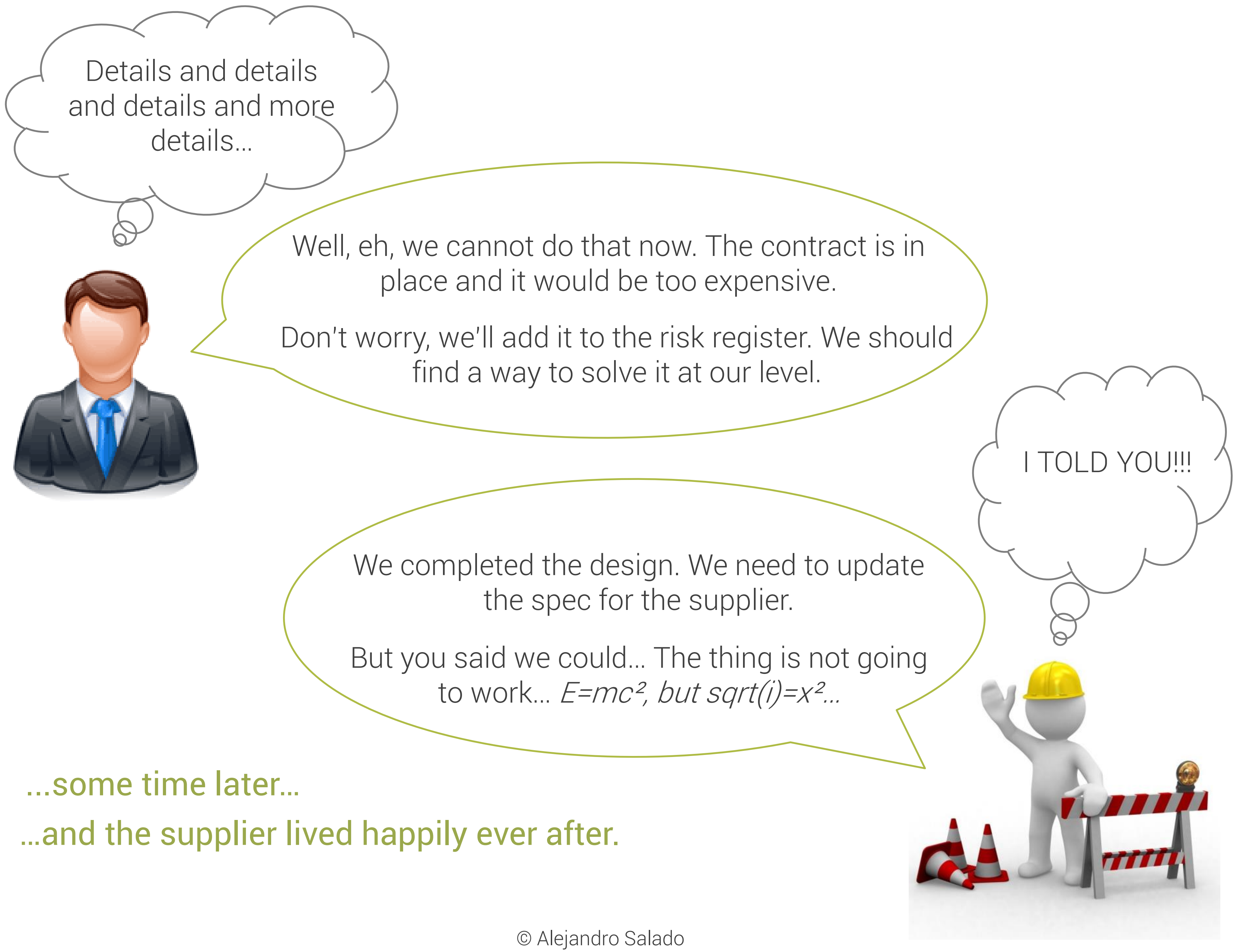
Don't worry, this is going to be only the first release. We can update it later, once the design matures...

How do these guys make decisions?

We can't, it isn't ready! Design is immature!

I tell you this is wrong. I'll give you the spec, but be aware of the risk we are taking...

Once upon a time, at the beginning of a project...


Details and details and details and more details…
Well, eh, we cannot do that now. The contract is in place and it would be too expensive.
Don't worry, we'll add it to the risk register. We should find a way to solve it at our level.
I TOLD YOU!!!
We completed the design. We need to update the spec for the supplier.
But you said we could… The thing is not going to work… *$E=mc^2$, but $sqrt(i)=x^2$…*

…some time later…

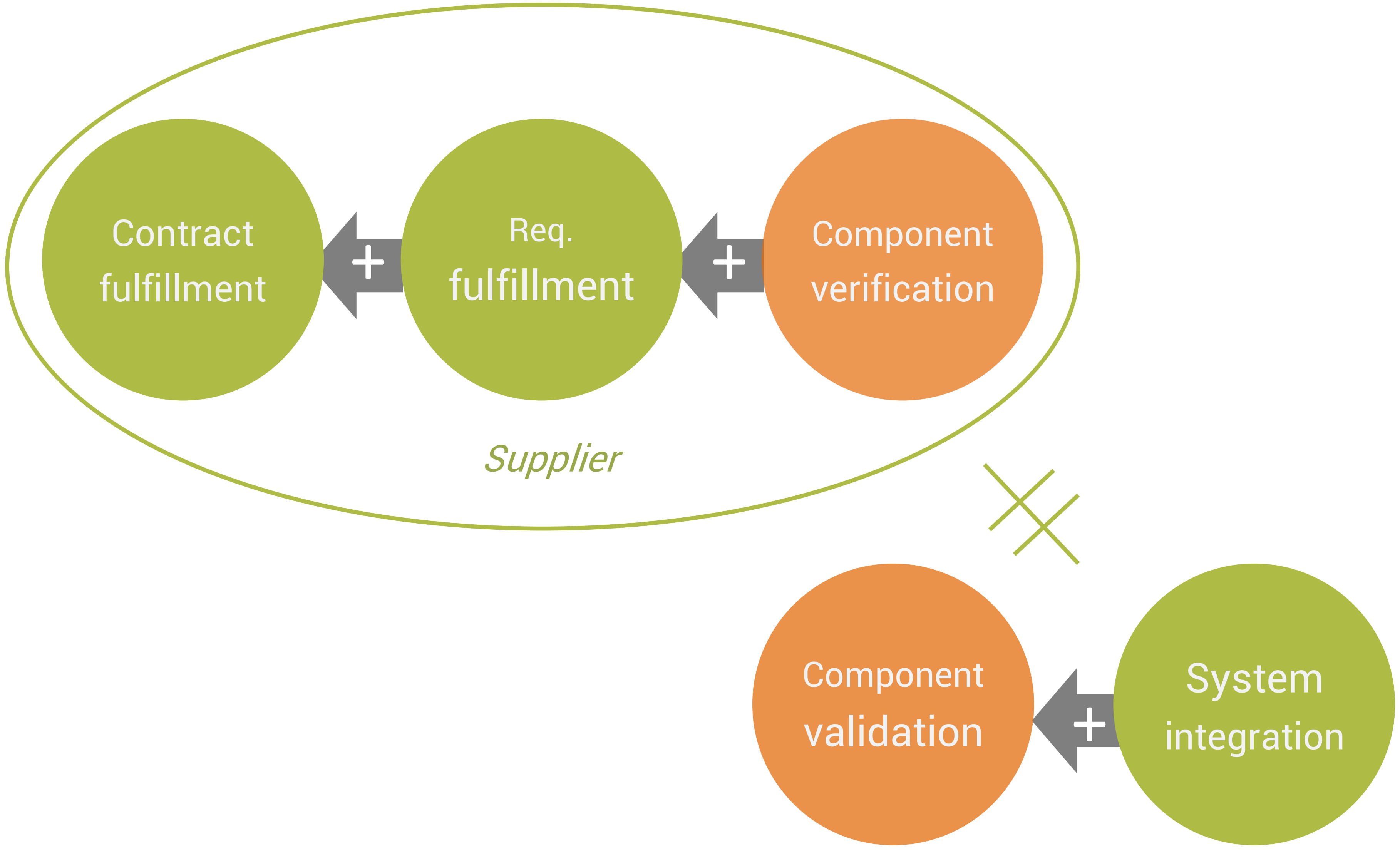
…and the supplier lived happily ever after.

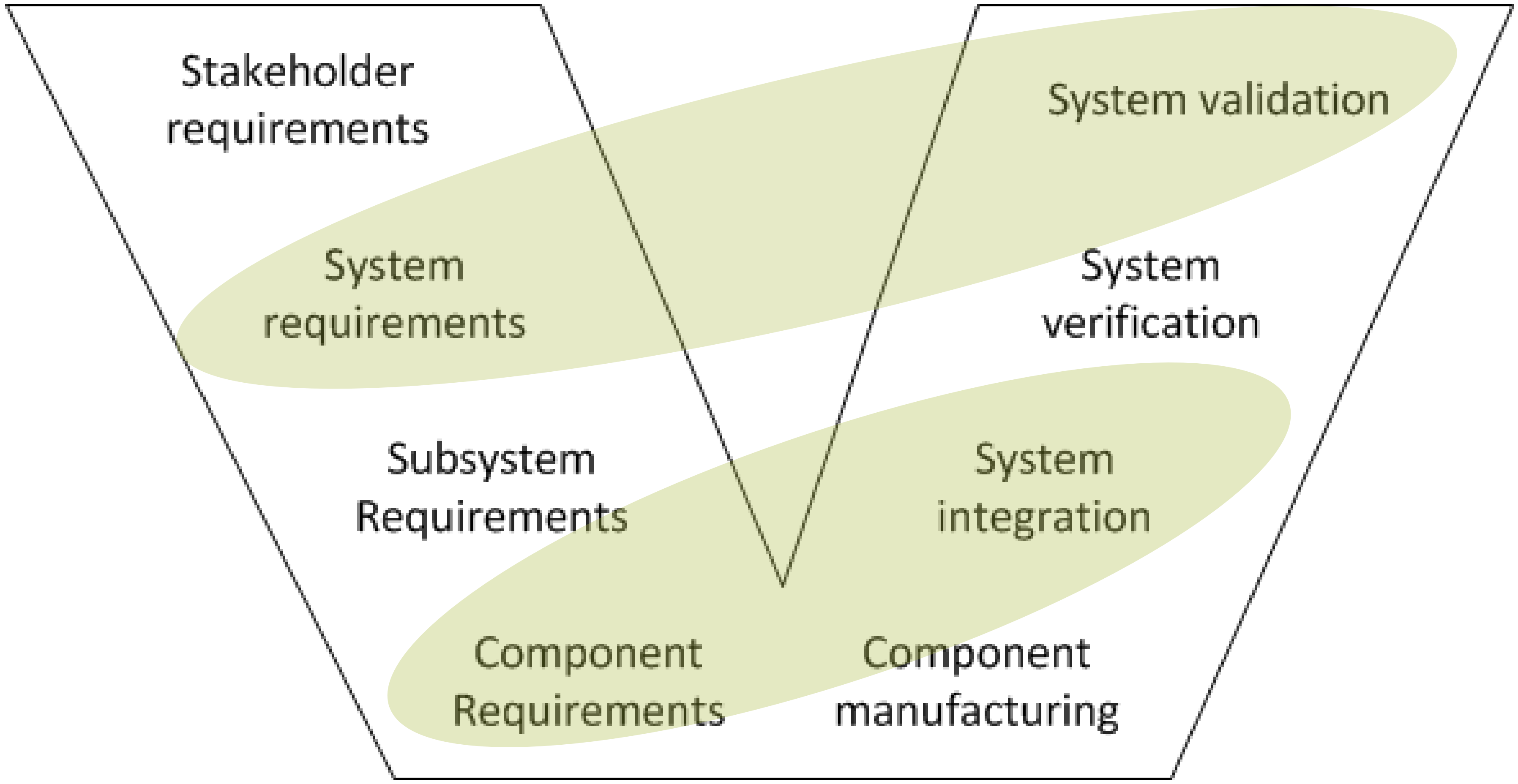
HAVING a problem IS NOT a problem.

PAYING a problem IS a problem.

How does your **CONTRACTUAL ARQUITECTURE** relate to your **SYSTEM ARCHITECTURES**?

Any crazy guy developing a contractual architecture? Really?


Contract fulfillment
+
Req. fulfillment
+
Component verification
Supplier
Component validation
+
System integration



Hierarchical? → NOT necessarily!

How do you DE-RISK validation while fulfilling a CONTRACT?

Ingredients

Requirements

+

Needs*

* When tightly coupled

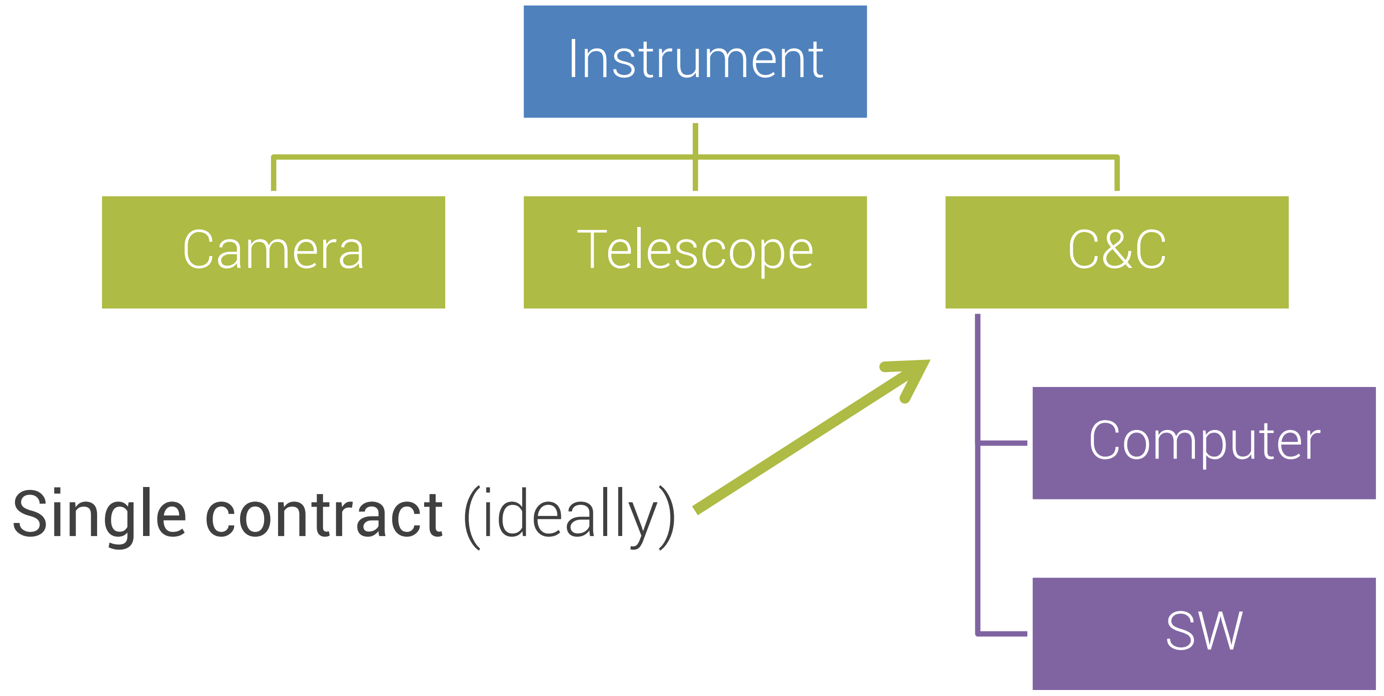
It **IS NOT** about **AVOIDING** problems…

… it IS **about** NOT PAYING for them!

SHARED payment milestones

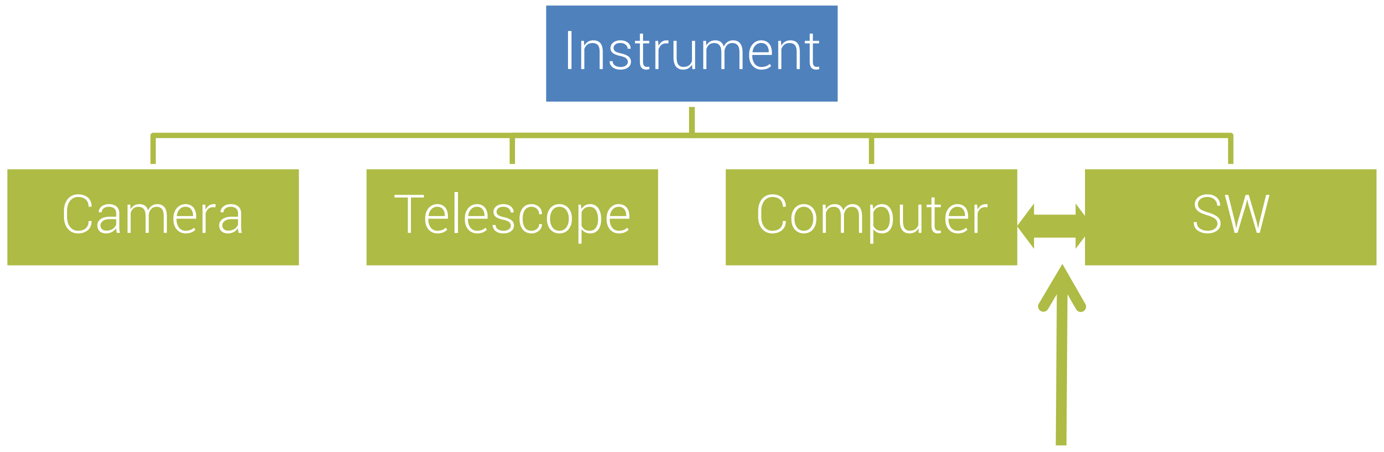
OVERCOMING

HW / SW integration
When coupling is tight



Because of geo-return…

Split of tight couples

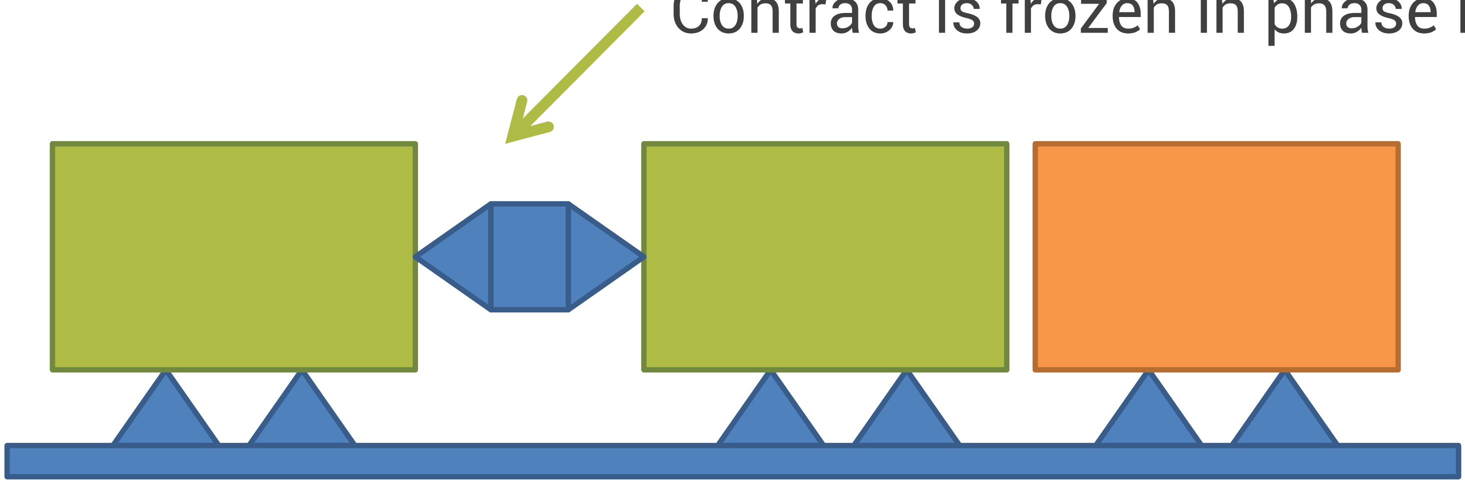


Link contract to a need

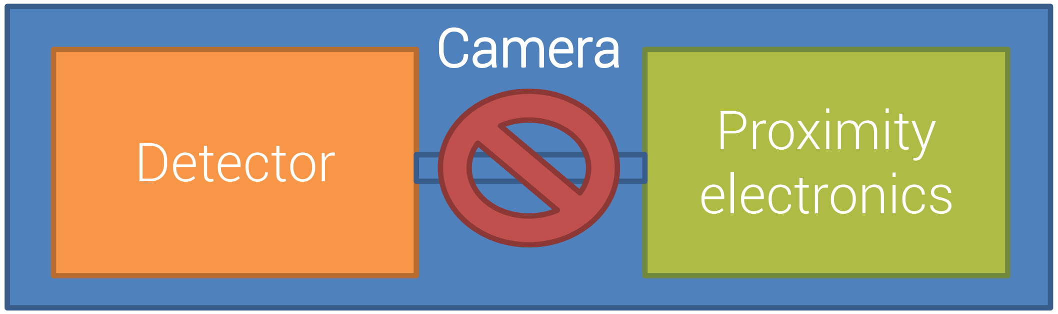
More expensive, but lower cost growth potential

Fulfill this… using this
The CFI leverage

Contract is frozen in phase B!

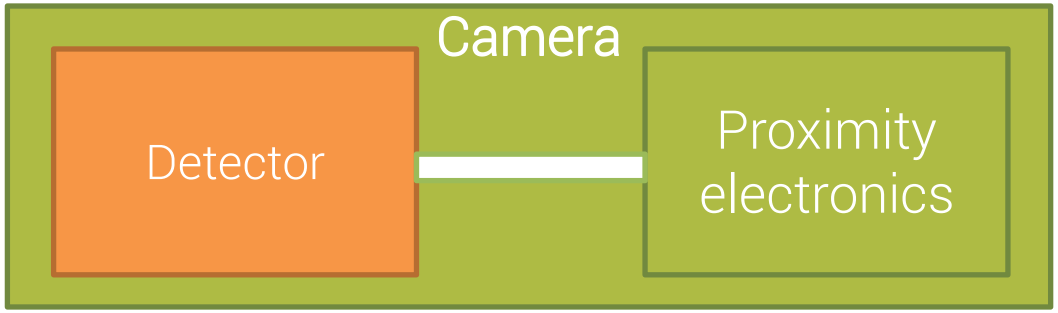


Not such a good partitioning & volatile design



Vs.

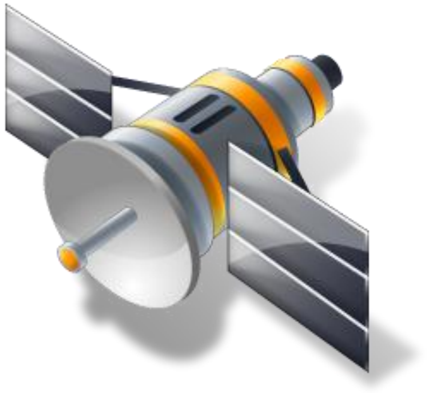
Isolate new development
Early procurement
Technology driver



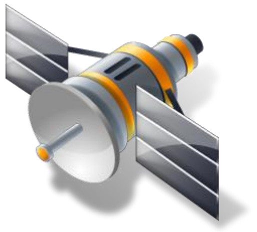
Difficult not robust interface

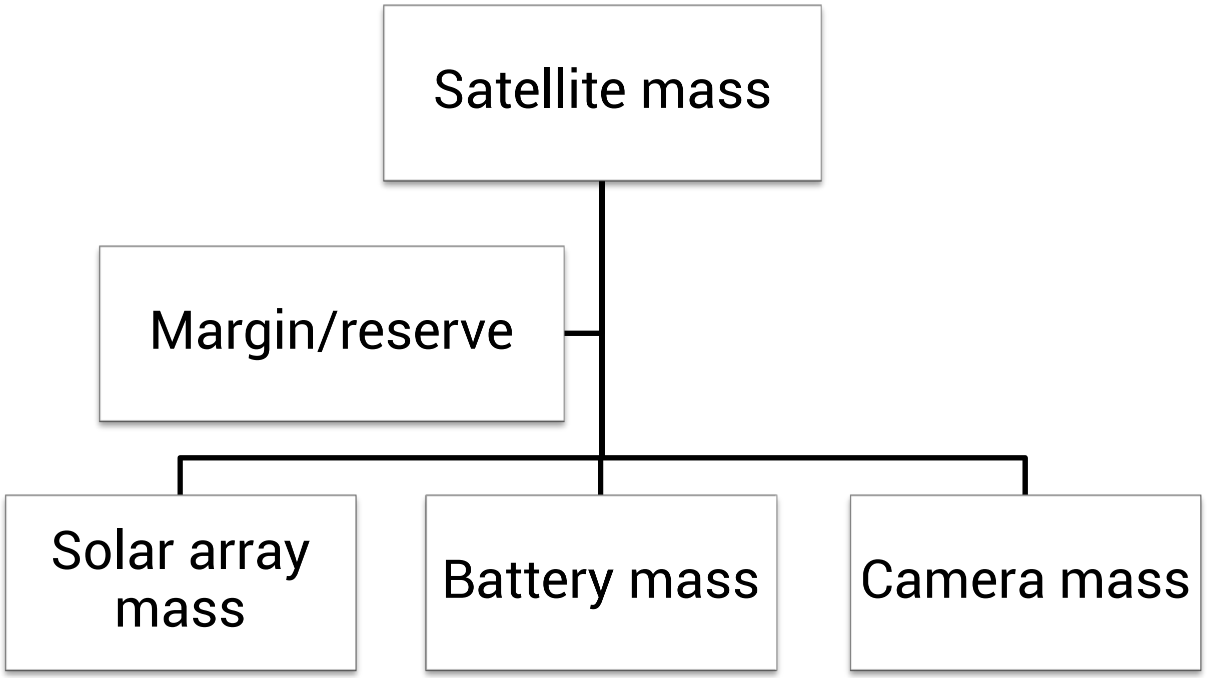
Requirements as commodities

Budgets in a trading market



Allocating resources

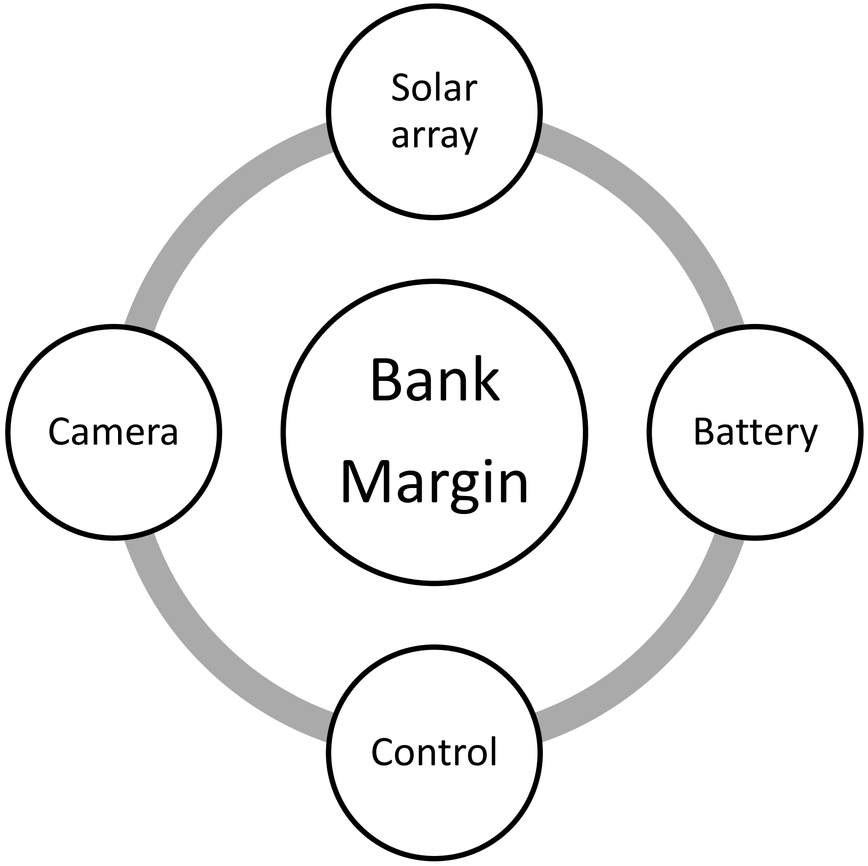




These are requirements!

Allocate budget

Exhaust allocations



These are NOT requirements!

Define and allocate commodities

Optimize for selling surplus

REFLECTION

Embrace uncertainty &
make it part of **your life**.

But narrow it!

Systems engineering proverb

CONTRACTS define your real architecture.

So DRIVE

your contractual architecture.

Systems engineering proverb

If you integrate them, you validate them.

If you do **not integrate** them,

they are **NOT VALIDATED**.

Systems engineering proverb

That one who
owns the requirements,
owns their validation.

Systems engineering proverb

MANGE **TAKK**

alejandro.salado@kayser-threde.com