10 Things You Must Know Before Designing Your Own Camera

Alex Fink
Panopteo

Panopteo LLC
Expert technical consulting

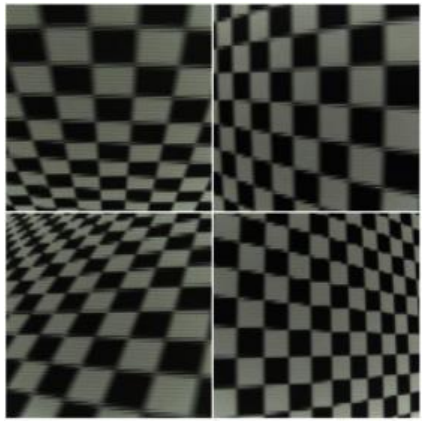
Things that AREN'T

Thing 1 - a camera is not a widget

Very few contract manufacturers know how to handle optics

Thing 1 - a camera is not a widget

Even fewer know how to do geometric calibrations…



Thing 1 - a camera is not a widget

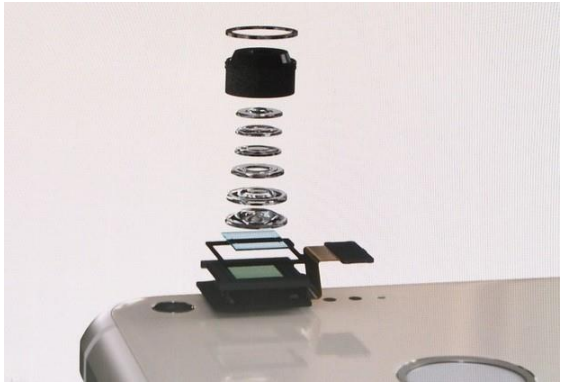
The more specialized skills you require, the fewer CMs will qualify

Thing 1 - a camera is not a widget

If you shop later (after you know your exact requirements)

You'll shop better

Thing 2 - a camera does not have ONE key component



What lens people think

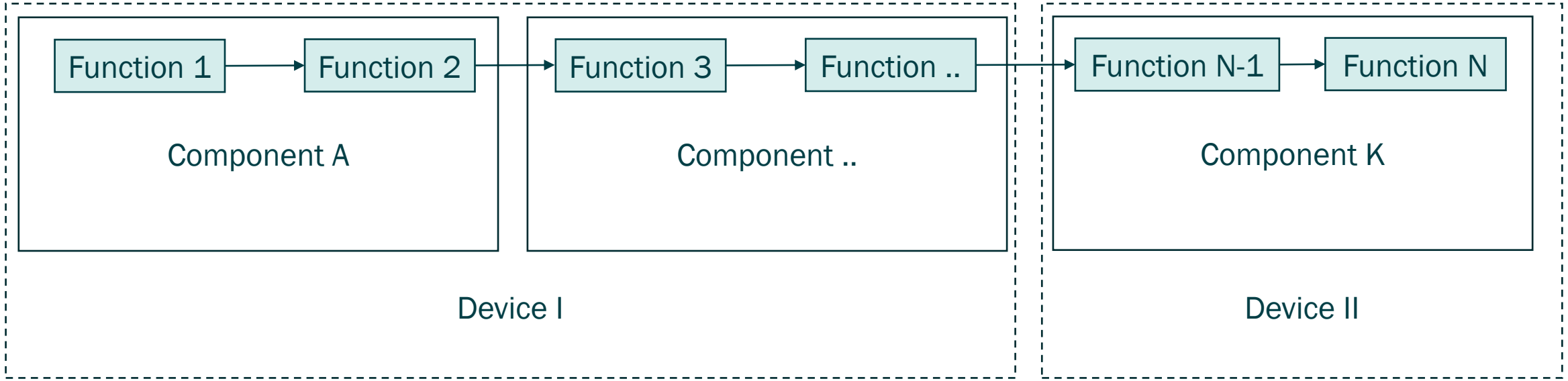
What SoC people think

What sensor people think

Thing 2 - a camera does not have ONE key component

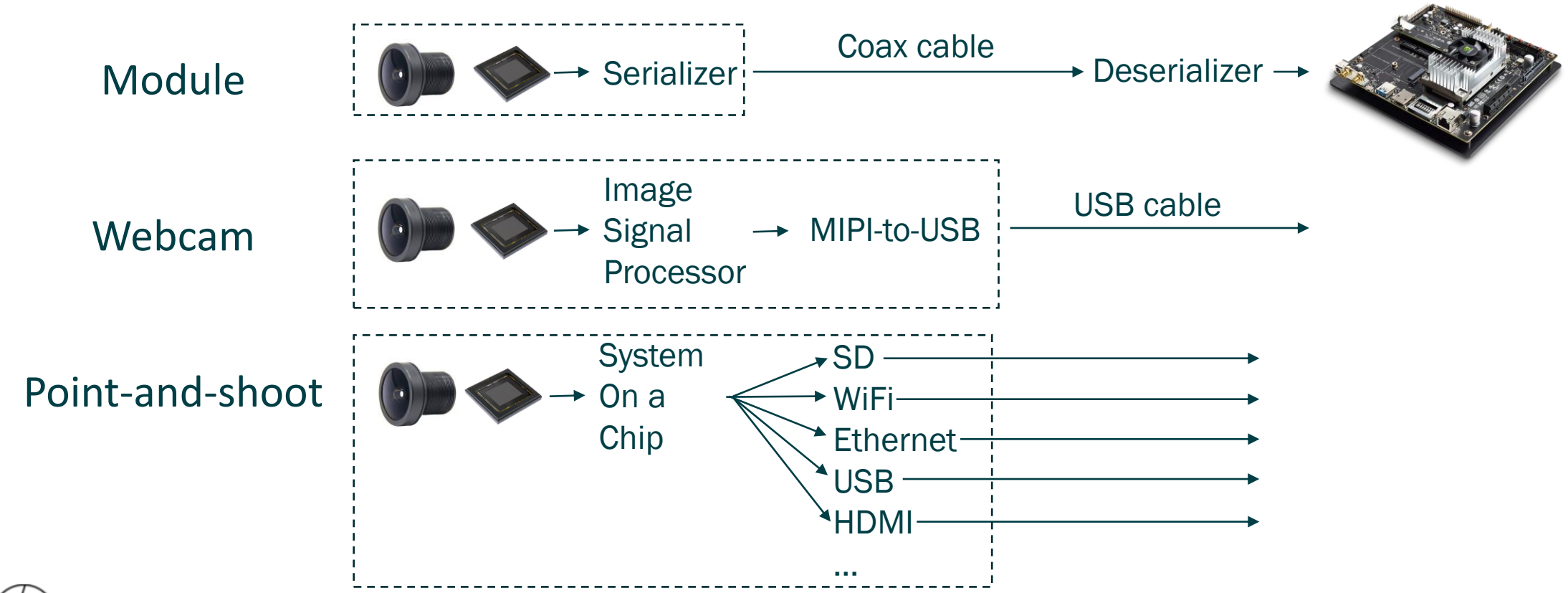
A univariate approach will not make a great camera

It’s better to ask -- what does the entire pipeline look like?



Thing 2 - a camera does not have ONE key component

And what do you mean by “camera”?



Thing 3 - computer vision development is not camera development

Scrums

Sprints

Frequent iteration

Fail fast!

But

Do these methods work when *compilation* takes 3-6 months?

Thing 3 - computer vision development is not camera development

State-of-the-art tech is better than old tech

But

What if the state-of-the-art hasn't shipped yet?

Do you want to be the beta-tester for your vendors' experiments?

Thing 3 - computer vision development is not camera development

The best spec is “as good as possible”

But

Can you select vendors and design partners to meet this spec?

What would you write in the SOWs?

Things that ARE

Thing 4 – good HW design starts from the end (and works backwards)

The camera has users (human or otherwise)

The users have use-cases

The use-cases can succeed or fail

What optical requirements will determine the outcome?

Thing 4 – good HW design starts from the end (and works backwards)

The product needs to ship on a particular date

The program has a budget

What architecture and components can be used to achieve the requirements -- on schedule and on budget?

Thing 4 – good HW design starts from the end (and works backwards)

Optics are a subset of physics

Most things are either feasible or not feasible

Doing the math is cheaper than building HW!

Thing 5 – platforms are sticky

Some choices are harder to change than others

Software engineers don't like it when the platform is swapped under their feet

but

The platform you prototype on might not be shippable

Thing 5 – platforms are sticky

Many of my customers start out on open-source HW

Many of my customers start out on off-the-shelf HW

Few of my customers ship products on either

Thing 5 – platforms are sticky

It’s best to make all prototypes on something that can be a mass-production platform

Things that OUGHT to be (done)

Panopteo LLC
Experi technical consulting

Thing 6 – outsource anything that isn't your core competence

Do you really need optics engineers in-house?

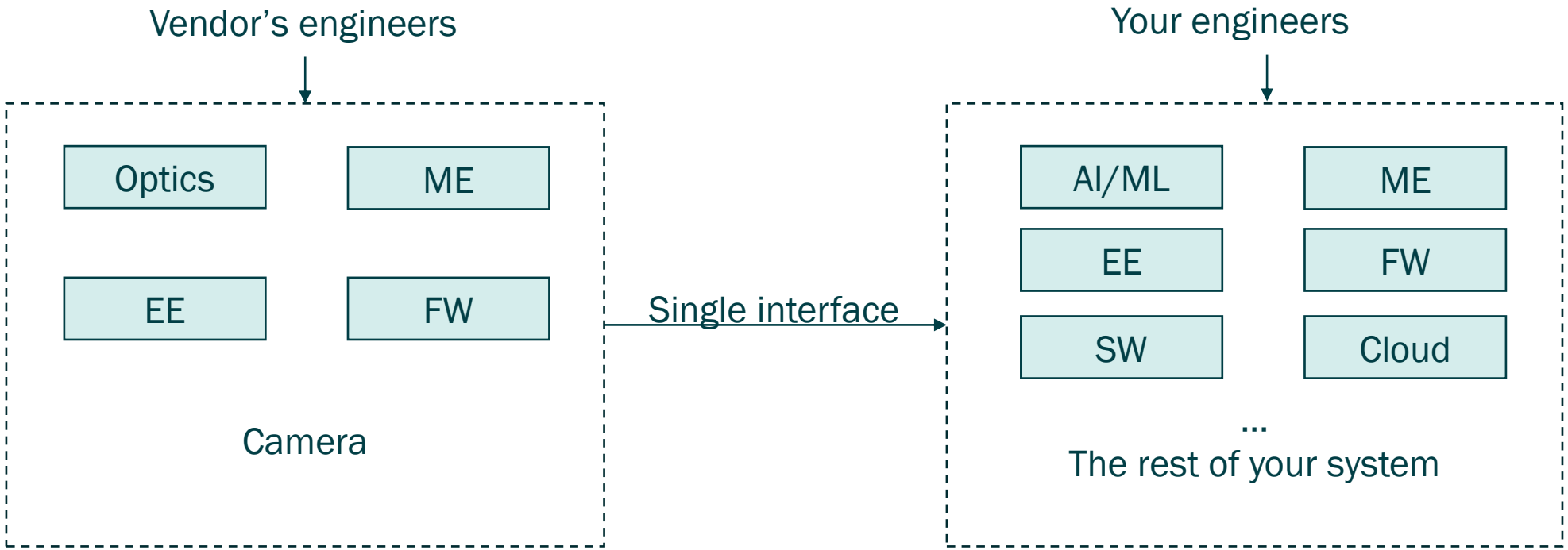
Thing 6 – outsource anything that isn't your core competence

Do you really need real-time firmware engineers in-house?

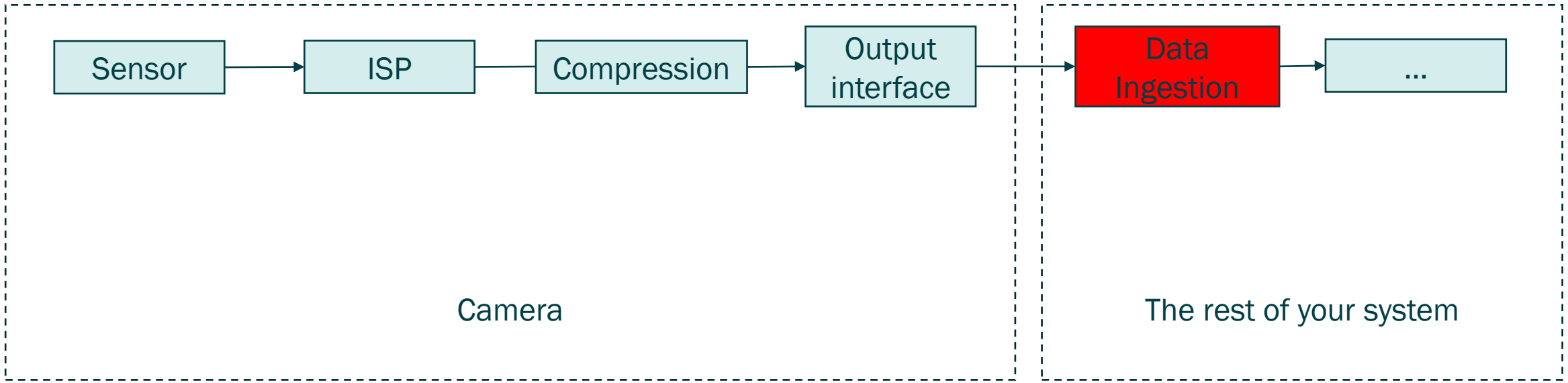
Thing 7 – set a clear line between your engineering and your vendors' engineering

If possible, separate on-camera work

from the rest of the system



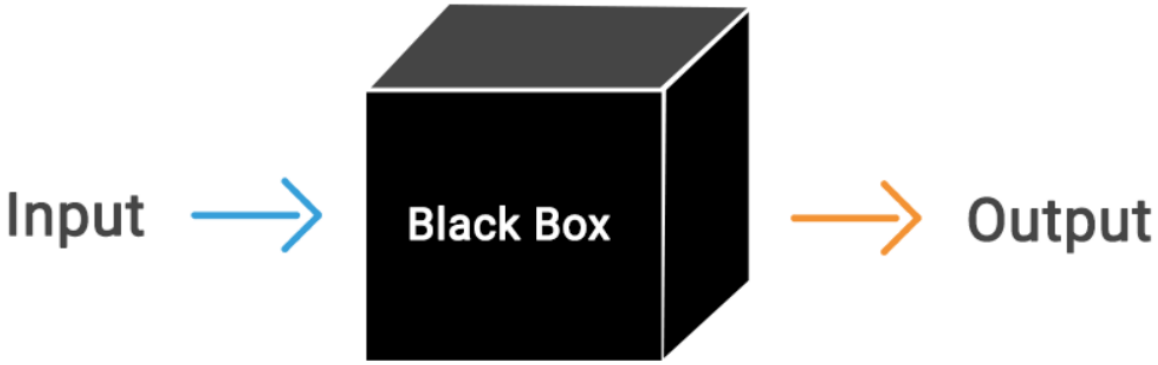
Thing 7 – set a clear line between your engineering and your vendors' engineering



Who is in charge of integrating every new HW or FW version of the camera?

Thing 8 – set clear acceptance criteria for vendors' deliverables

If the camera were a standalone device, how would you test it?



How do your real-world criteria map to camera criteria?

Thing 8 – set clear acceptance criteria for vendors' deliverables

Is there a list of functions and test-cases?

Can you reject deliverables?

Can the vendor disagree?

Thing 9 – test your own image quality (even if the vendor already tested it)

What does "good image quality" mean to you?

Thing 9 – test your own image quality (even if the vendor already tested it)

2021 embedded VISION summit

In-lab tuning vs real-world scenarios

Thing 10 – plan for V2.0 (before V1.0 is done)

Where do all the great ideas go?

Thing 10 – plan for V2.0 (before V1.0 is done)

Your engineers should trust that shelved features aren't discarded features

Thing 10 – plan for V2.0 (before V1.0 is done)

It helps to launch V2.0 pre-planning

In parallel to V1.0 development

Recap

1. A camera is not a widget
2. A camera does not have ONE key component
3. Computer vision development is not camera development
4. Good HW design starts from the end (and works backwards)
5. Platforms are sticky
6. Outsource anything that isn't your core competence
7. Set a clear line between your engineering and your vendors' engineering
8. Set clear acceptance criteria for vendors' deliverables
9. Test your own image quality (even if the vendor already tested it)
10. Plan for V2.0 (before V1.0 is done)

Thank you!

Panopteo LLC
Experi technical consulting

References

Panopteo's website:

http://panopteo.com/

My email:

alex@panopteo.com

My linkedin:

https://www.linkedin.com/in/temuchin43/