Génie logiciel pour la conception d'un Système d'Information

CSC4521

Voie d'Approfondissement
Intégration et Déploiement de Systèmes d'Information
(VAP DSI)

FunctionalDesign

http://jpaulgibson.synology.me/~jpaulgibson/TSP/Teaching/CSC4521/

…/CSC4521/CSC4521-FunctionalDesign.pdf


NICE DESIGN.... BUT YOU HAVEN'T QUITE MASTERED THE TECHNOLOGY!
search ID: mmo0009
© Original Artist
Reproduction rights obtainable from
www.CartoonStock.com

Some Software Design Quotes

The designer of a new kind of system must participate fully in the implementation.
—Donald E. Knuth

... the designer of a new system must not only be the implementor and the first large-scale user; the designer should also write the first user manual. ... If I had not participated fully in all these activities, literally hundreds of improvements would never have been made, because I would never have thought of them or perceived why they were important.
—Donald E. Knuth

Some Software Design Quotes

There are two ways of constructing a software design. One way is to make it so simple that there are obviously no deficiencies. And the other way is to make it so complicated that there are no obvious deficiencies.
—C.A.R. Hoare

It's hard to read through a book on the principles of magic without glancing at the cover periodically to make sure it isn't a book on software design.
—Bruce Tognazzini

Some Software Design Quotes

Design is the art of separation, grouping, abstraction, and hiding. The fulcrum of design decisions is change. Separate those things that change for different reasons. Group together those things that change for the same reason.
—Robert Martin

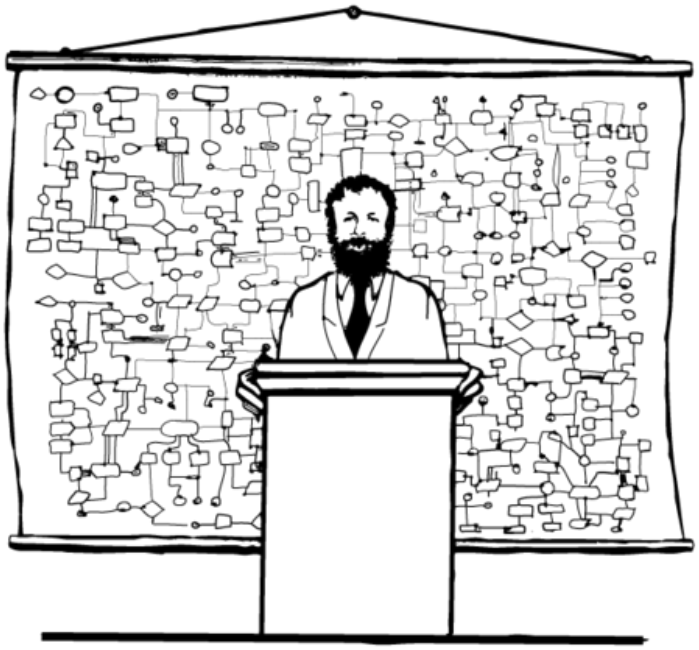
The hardest part of design ... is keeping features out.
—Donald Norman

Some Software Design Quotes

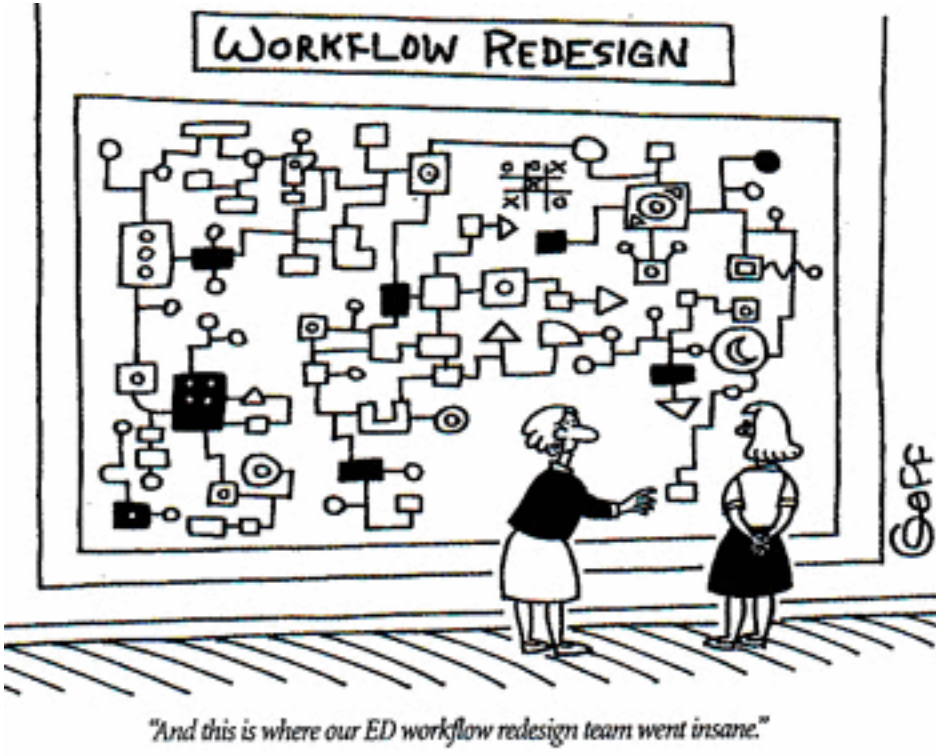
Design and programming are human activities; forget that and all is lost.
—Bjarne Stroustrup

The tragedy of our time is that we've got it backwards, we've learned to love techniques and use people.
—Herb Kelleher

How To Judge If A Design Is Good?



"Now that you have an overview of the system, we're ready for a little more detail"



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

http://timbroder.github.io/jira-presentation/

Some Suggested Reading

On the Criteria To Be Used in Decomposing Systems into Modules, Parnas, 1972

A Rational Design Process: How and Why to Fake It, Parnas and Clements, 1986

A field study of the software design process for large systems, Curtis, B. and Krasner, H. and Iscoe, N.,1988

What is Software Design?, Jack W. Reeves, 1992

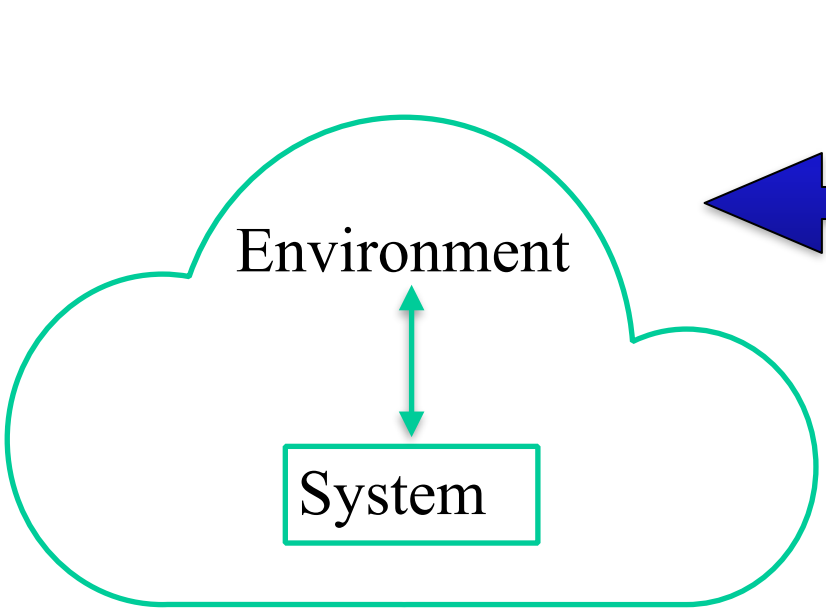
Bad smells in code, Beck and Fowler, 1999

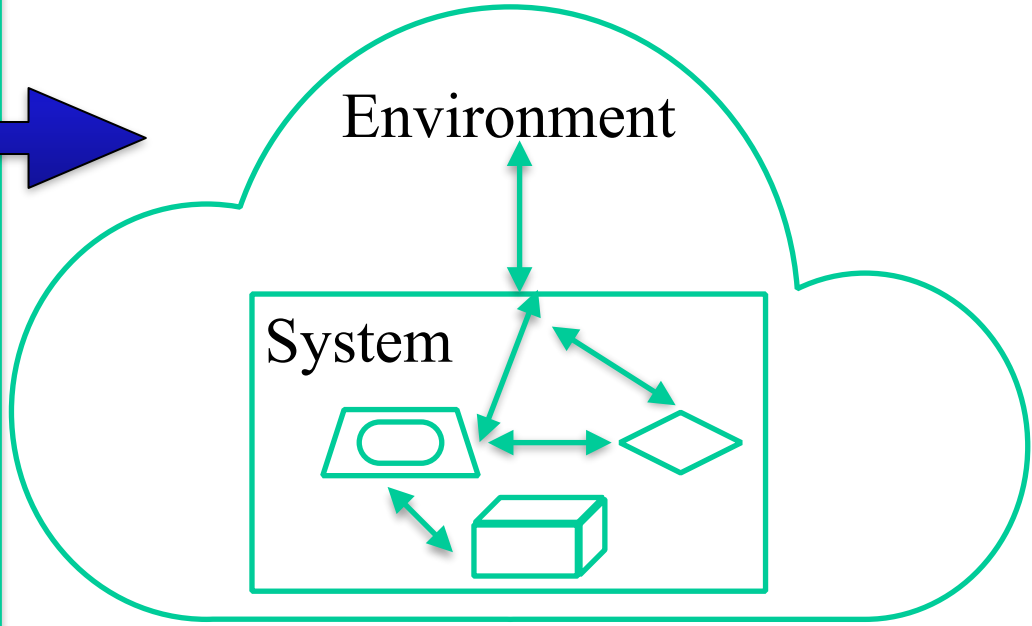
The risks of stopping too soon, Parnas, 2011

Some Useful Links

- What does a Functional Design have to offer?, ITpedia.
- WHAT IS A FUNCTIONAL DESIGN SPECIFICATION (FDS)?, RealPars.
- What is a Functional Specification Document?, Essential Designs.

The **Functional Design** must be coherent with the **Functional Requirements**





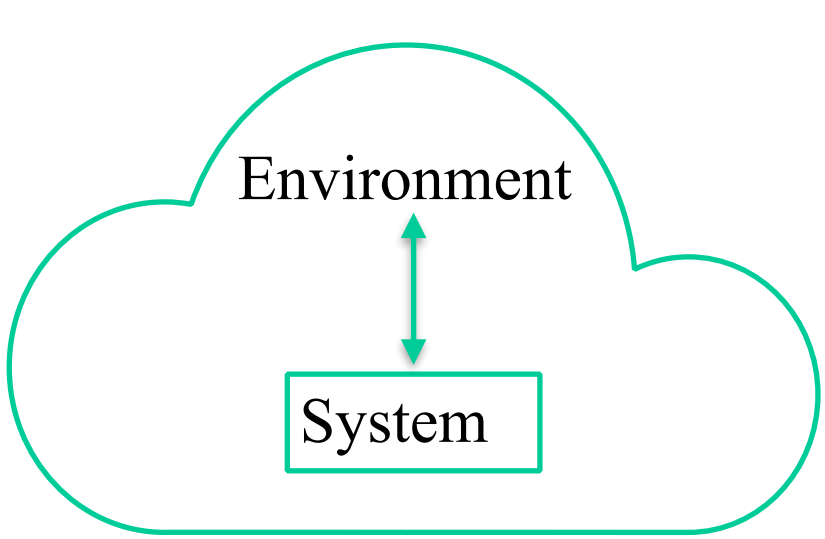
Functional Requirements

Structured model of interactions between system and its environment (including users)

Functional Design

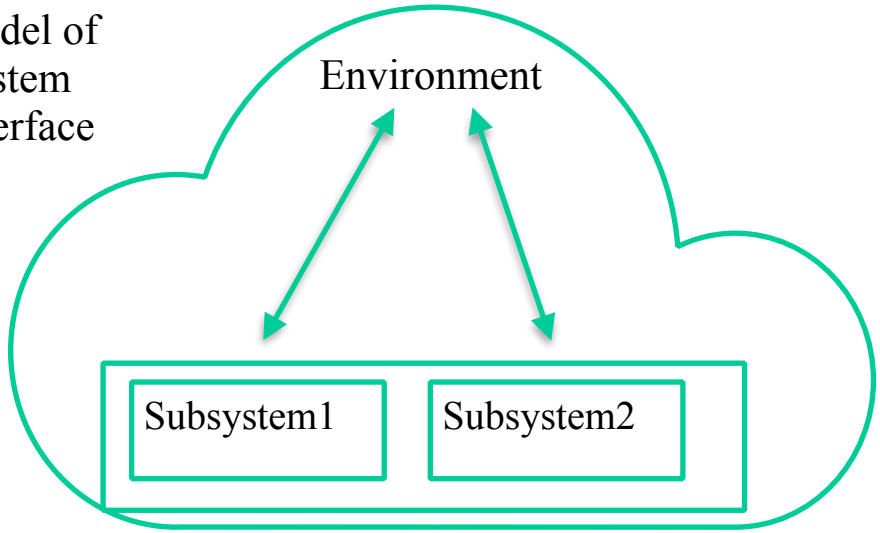
Structured model of interactions between system components in response to interaction with the environment

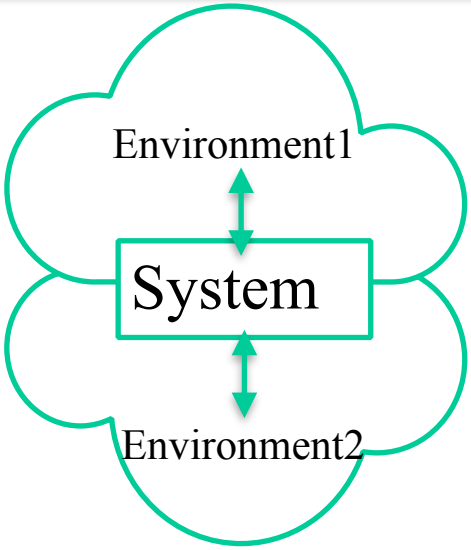
Functional Requirements can be structured/distributed



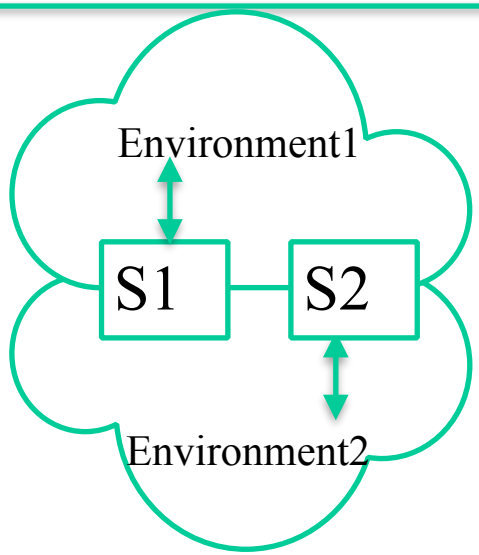
Flat model of Environment and System Interfaces

Structured model of System interface





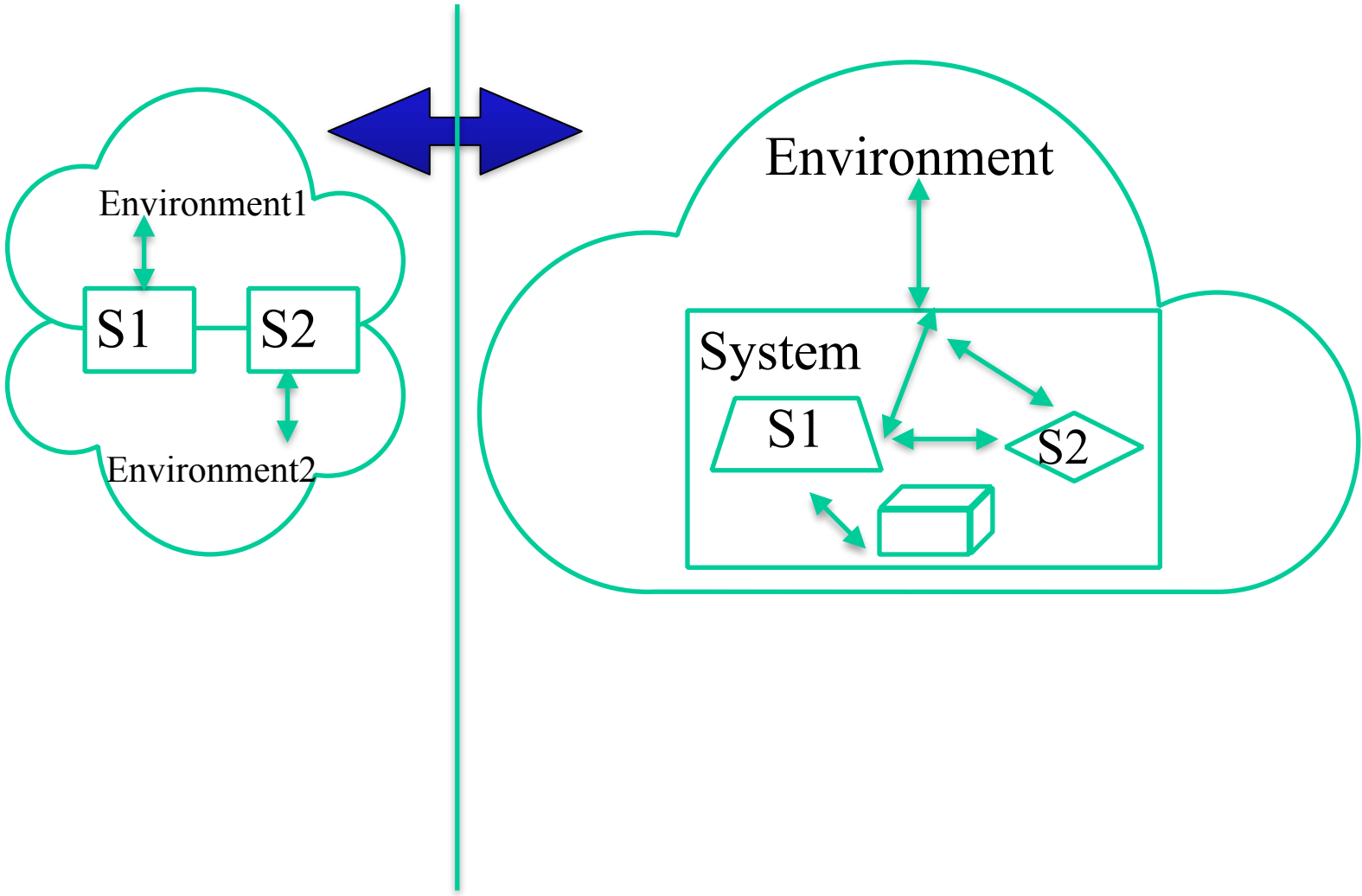
Structured model of Environment interface



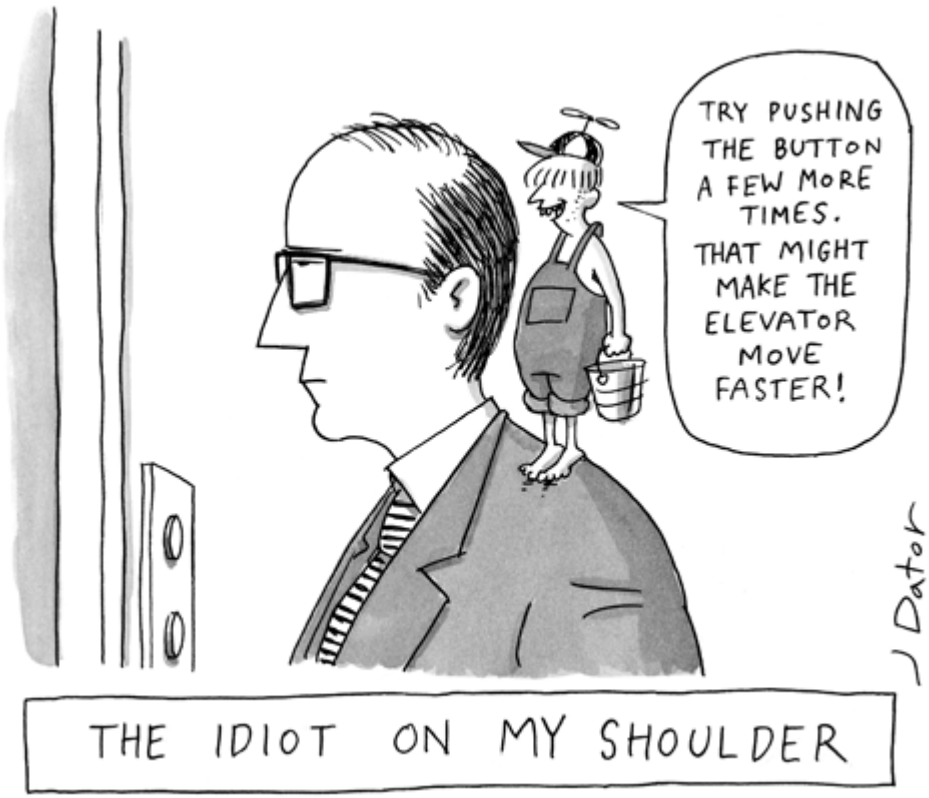
Structured model of Environment and System Interfaces

Structure in requirements may match structure in design

Structured model of Environment and System Interfaces as part of requirements



Lets Try Out Some Design With Our Lift/Elevator



Propose a functional design for meeting your **lift** requirements

Lets Try Out Some Design With Our Lift/Elevator

This can be done:

- top-down (identify components from use cases), or
- bottom-up (construct use cases from components)
- inside-out (patterns between use cases and components)

What are the main components of our system?

Sensors/motors/doors etc…

What would we do if we start by looking at the doors?

Let us consider the **door** requirements and design

Requirement -"*doors only open if lift at rest at a floor*"

Design - we need to model the **state** of a door (at a good level of abstraction) in order to verify the requirement

Door Design1 - state is a simple boolean (open or closed)
Door Design2 - state is a simple boolean (open and not open)
Door Design3 - state is a simple boolean (not closed and closed)
Door Design4 - state is an enumeration - {open, partially open, closed}
Door Design5 - state is an enumeration - {open, opening, closing, closed}
Door Design6 - state is a real - distance apart in metres
derived attributes - closed <=> distance = 0
open <=> distance > 0

etc …

Let us consider the **door** requirements and design

Question - does a door need to know where it is?

In the elevator?
On a floor ?
On which floor?

Question - Do different **locations** require different behaviour? If so, do we need different types/classes of door?

Question - does a door need to know if it is **blocked**? Is this a state attribute of the door or a property of the system that can be derived (by looking at the value of a sensor, perhaps?) If we have a sensor for detecting blockages, then is it a part/component of the door?

What about the **Door API** ? - reading the state, changing the state

Question - are the state attributes visible to the environment, if so to all users or some users?

Question - what interface is offered to permit state changes -

logical “buttons” can have values on/off

For example, we require a **button** to open/close the door. Is this button visible to the environment? Internally what can see/use it?

Questions - is this button part of the door or is this button a separate component that communicates with the door. If it is a component then is the communication synchronous/asynchronous, direct/indirect? What is the data that is shared during the communication (if any). Is the communication secure/encrypted? Is there a handshake? Is it critical? What are the exceptions to be handled?

SPL - a family of lift systems

- Number of floors is configured at compile time
- Number of lifts is configured at deploy time
- Number of lifts operational can be reduced after deployment
- The lift controller logic can be hot-plugged (while lift is fully operational)

What ‘extra’ optional features can you think of?

Lets Try Out Some Design With Our Lift/Elevator

Functional Architecture Issues:

- Controller(s) - centralised/distributed/hybrid
- What is controlled (motors?)
- What functions are calculated (where)?
- What data is required by the functions?
- Where is data stored?
- What are functional dependencies?

Lets Try Out Some Design With Our Lift/Elevator

Functional Architecture Issues:

- Controller(s) - centralised/distributed/hybrid
- What is controlled (motors?)
- What functions are calculated (where)?
- What data is required by the functions?
- Where is data stored?
- What are functional dependencies?

Verifying Design against Requirements

TO DO -

Complete your functional lift design

How do you know if the design is functionally correct?