think flow

GROW BY FLOW

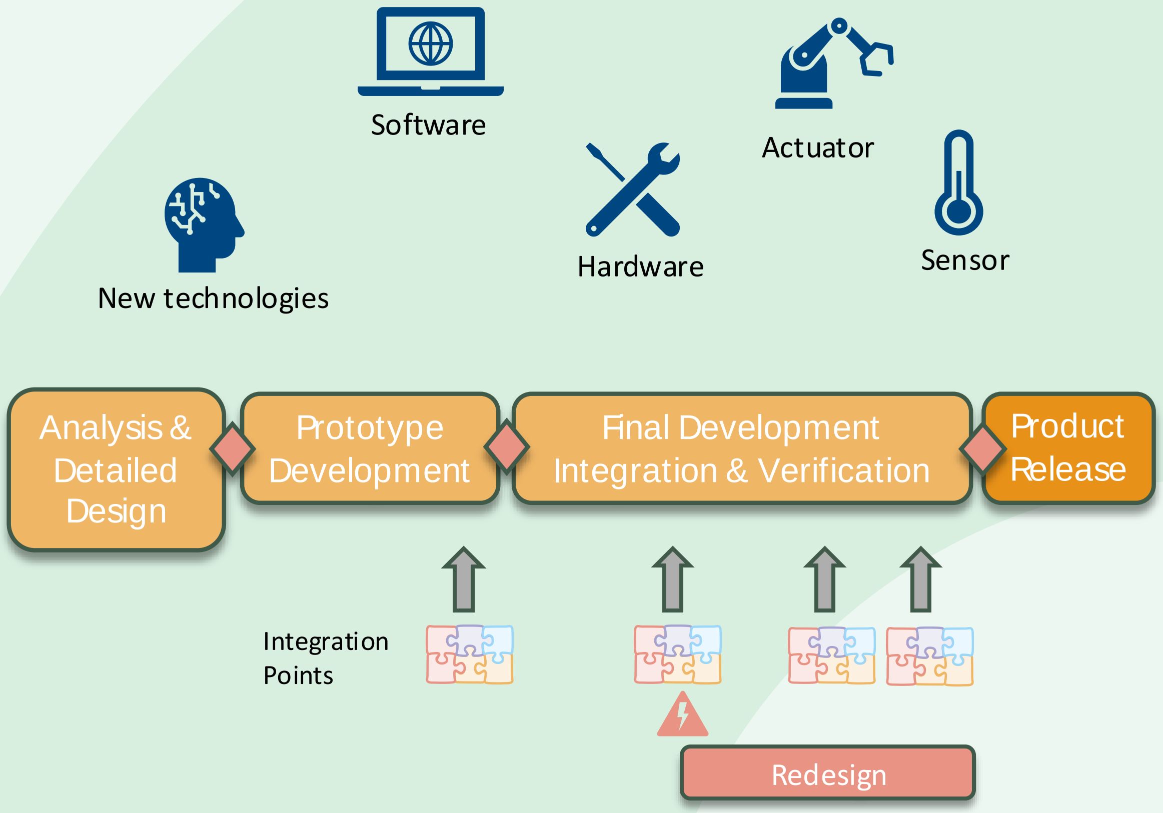
Advancing Hardware Development: A Learning-Centric Approach to Drive Innovation

2025-01-09

Dirk Holste

Challenges in product development

- Systems become more complex, interconnected and autonomous
- New technologies needs to be integrated faster
- Detected problems force unplanned redesign loops putting project success at risk



What is the terrain we operate in?

Clear Path

Improve

Known area

Find path

Experiment

Detect Problems

Learn

Innovate

Verification Mindset: No room for failure

- The purpose of tests are to verify that the system works as expected
- Failures are seen as negative and should be avoided
- This restricts the pace to learn and innovate

Learning based design – Continuous Prototyping

- Product development, is highly dependent on what needs to be learned
- Shift in the design process
  - Focus on obtaining knowledge at early stages
  - Keep design freedom and avoid early design choices

Change of Mindset

Design then test

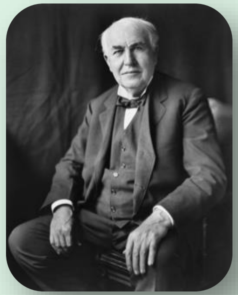
Learn then Design

Love failure & deviations

- Embrace failure and deviations to learn and progress faster
- Aim is to fail as inexpensively and early as possible
- Be guided by:
  - Is it safe enough to try?
  - What can we learn?

**““I have not failed.
I've just found 10,000 ways
that won't work.”**

Thomas A. Edison



Embrace failure – Amazon Fire smartphone

**“If you think that’s a big failure,
we’re working on much bigger failures right now
and I am not kidding.
Some of them are going to make
the Fire Phone look like a tiny little blip.”
- Jeff Bezos**

Enablers to drive innovation

Modularization
- Enables independent development, learning and innovation

Planned learning cycles
- Iterate and integrate
- Provides feedback and learning loops

Value based slicing
- Ensures small learning increments

think flow

Modularization

- Enables independent development and learning

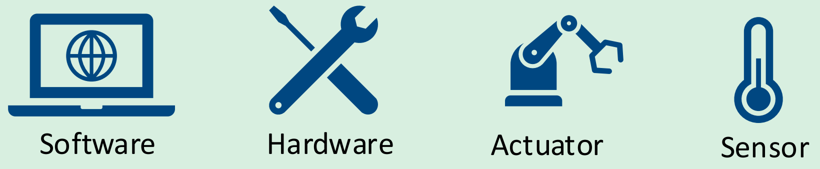
Component / Module

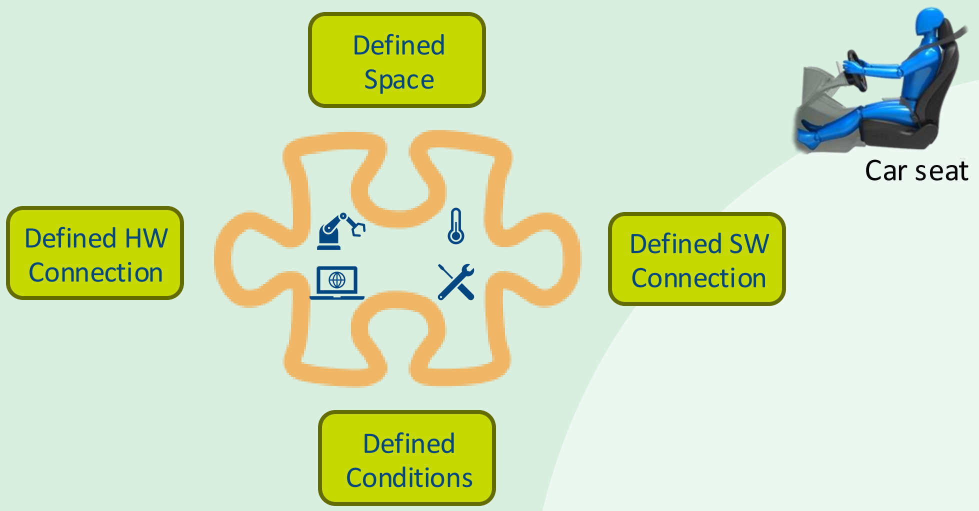
Component

- **Individual part or element** of a system that **contributes to the system's functionality**
- Each component typically has a specific role and interacts with other components to perform a function

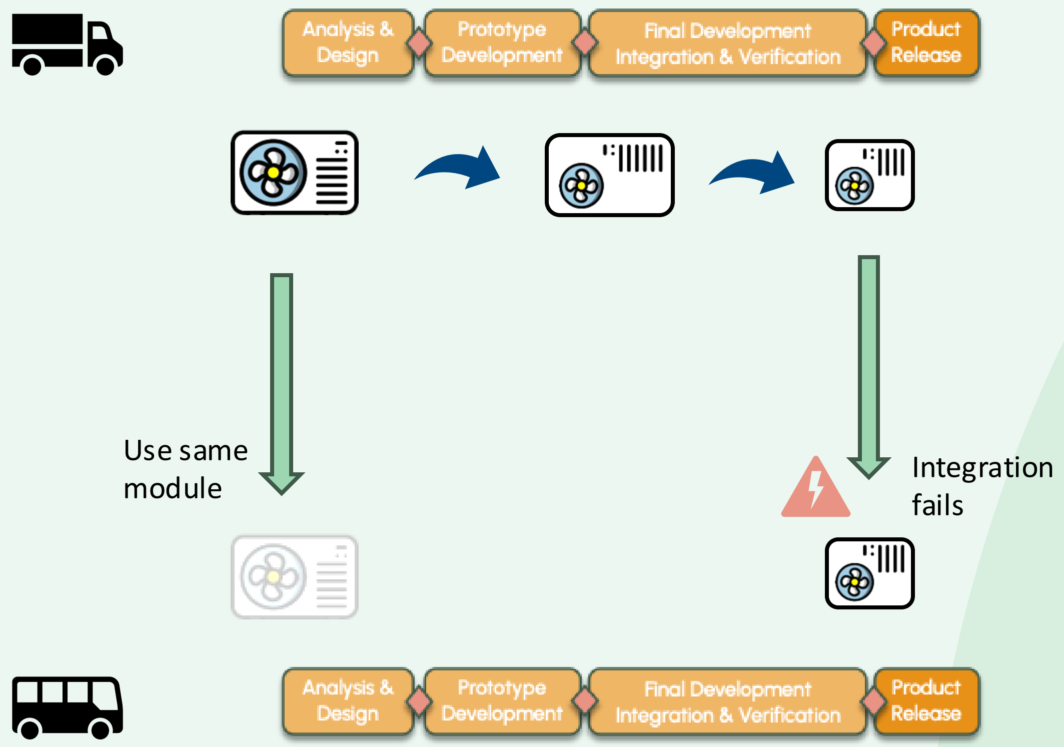
Module

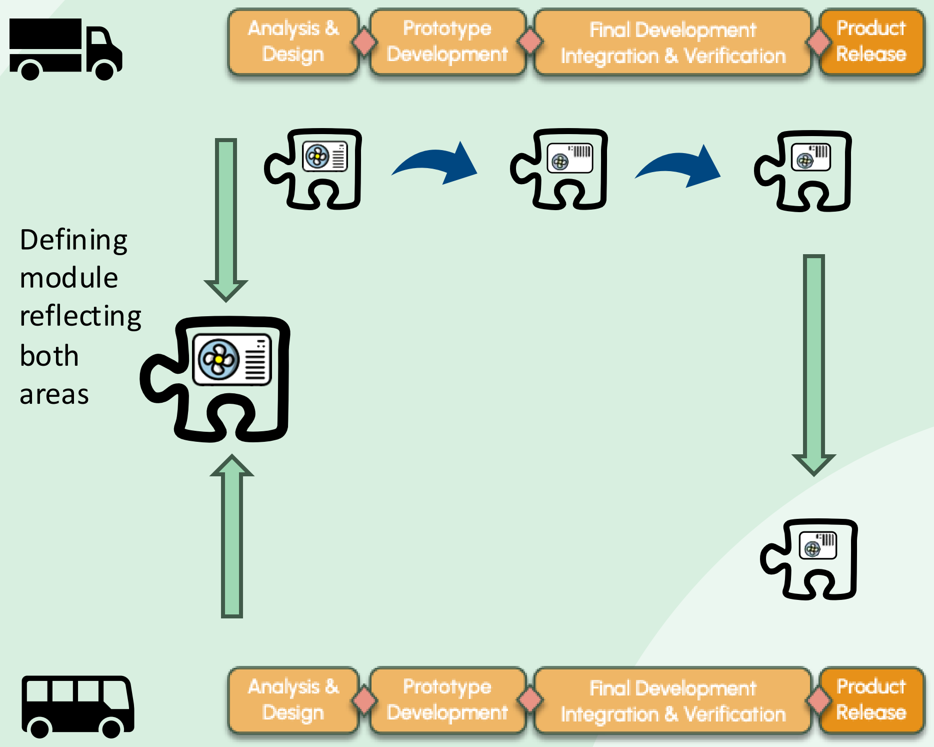
- **Self-contained unit** within a larger system that encapsulates a specific **(sub-) function or task**
- Designed to be relatively independent, with clearly defined interfaces for interaction with other modules
- Allow for flexibility, scalability, and customization within a system architecture
- It is NOT a concept that describes the overarching design idea or blueprint





Modules create development freedom

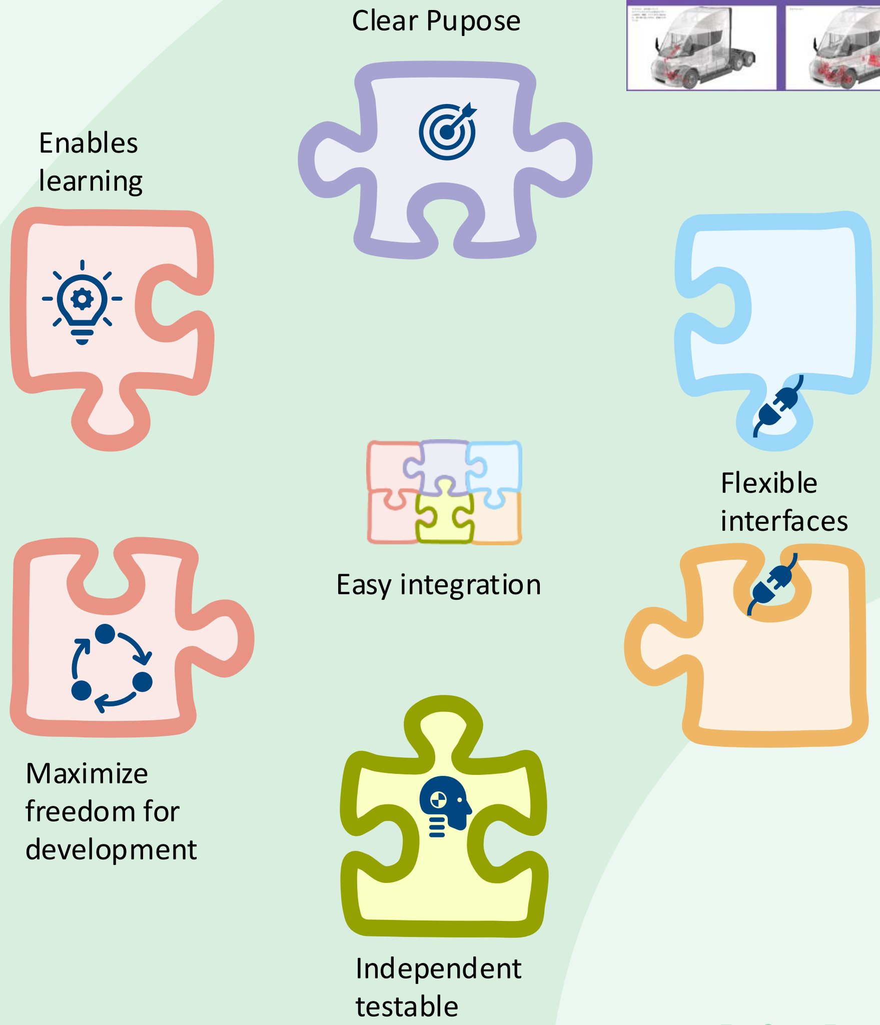



Analysis & Design
Prototype Development
Final Development Integration & Verification
Product Release
Defining module reflecting both areas
Analysis & Design
Prototype Development
Final Development Integration & Verification
Product Release

Module Architecture

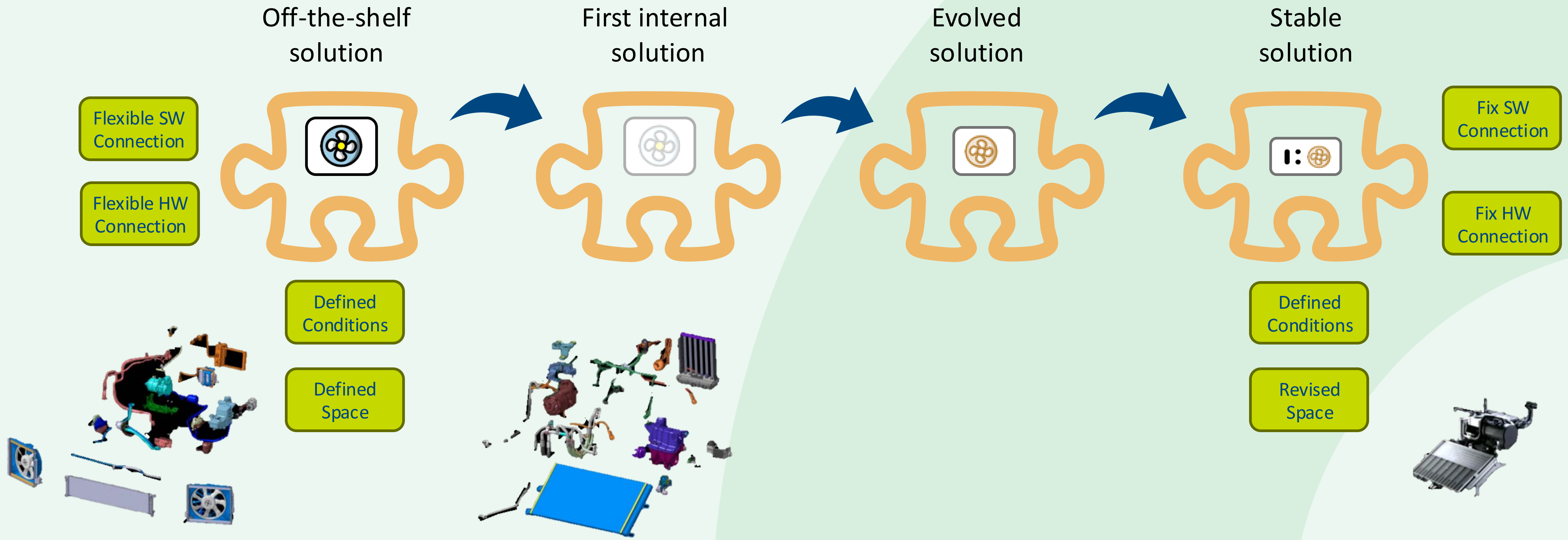
- Architecture is balancing many things
- Aim is to have losely coupled modules that:
  - Have clear purpose and link to product attributes
  - Create development freedom
  - Enable learning & innovation
  - Ensure easy integration

- Key insights:
  - Adaptability is your first concern
  - Cost of change is the second



Car manufacturer

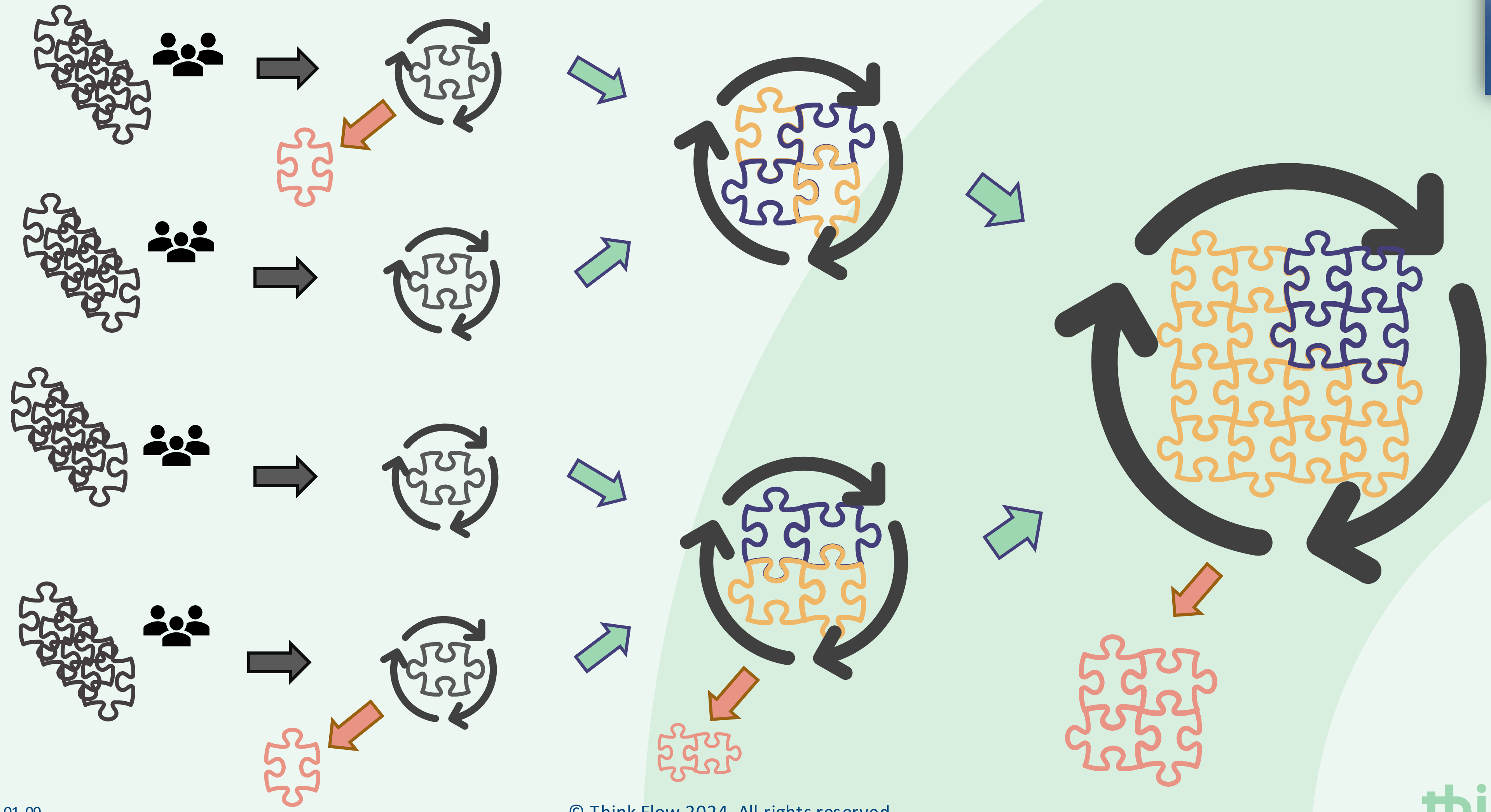
Modules enable innovation



Planned learning cycles

- Iterate and integrate
- Provide feedback and learning loops

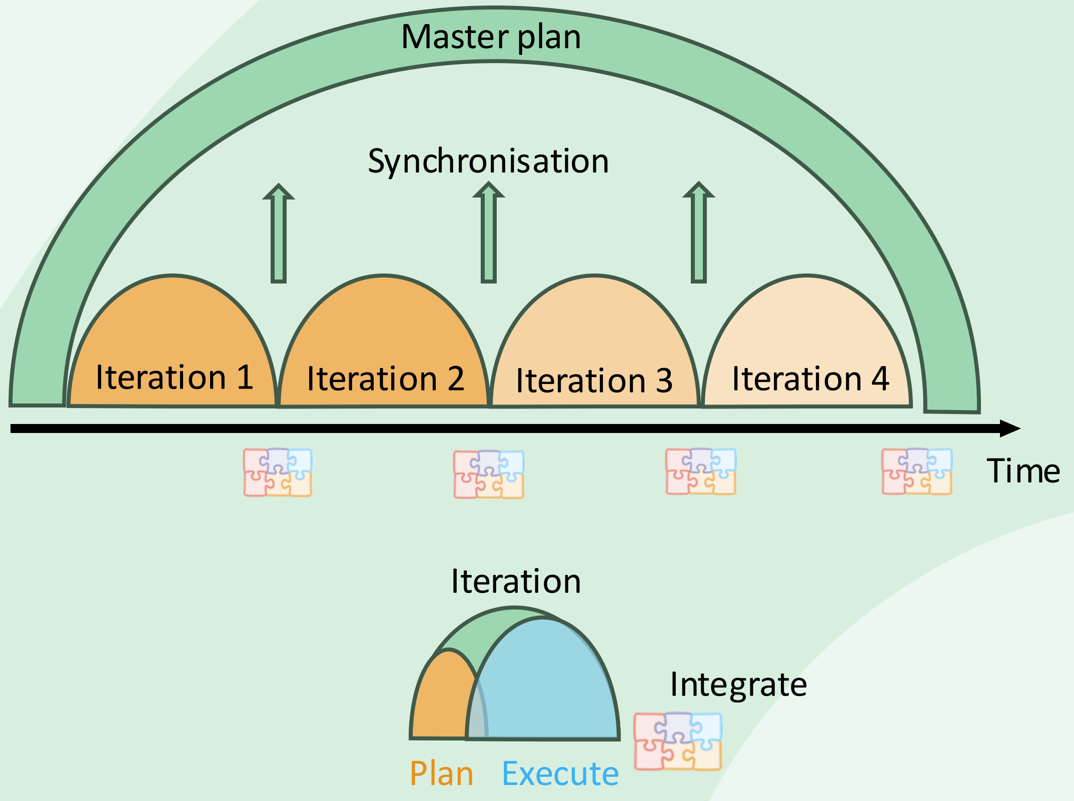
Iterate, integrate and evaluate on all levels



Automated integration & test

Rolling wave planning

- Two-level rolling planning
  - High level overall plan (master plan) that is roughly planned to the end of the program
  - Several iterations that consist of a planning and execution phase
- Learning from the iteration is synchronized with the higher-level plan and adjusted as necessary
- Advantages are:
  - Ability to adapt to changes as the programme progresses, as we plan to re-plan
  - More information will be achieved before doing detailed planning



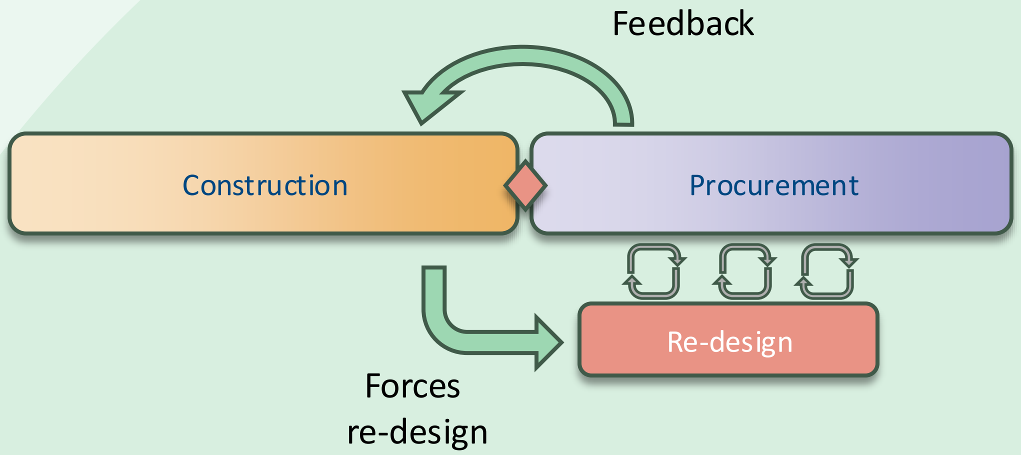
Experience from e-propulsion bus

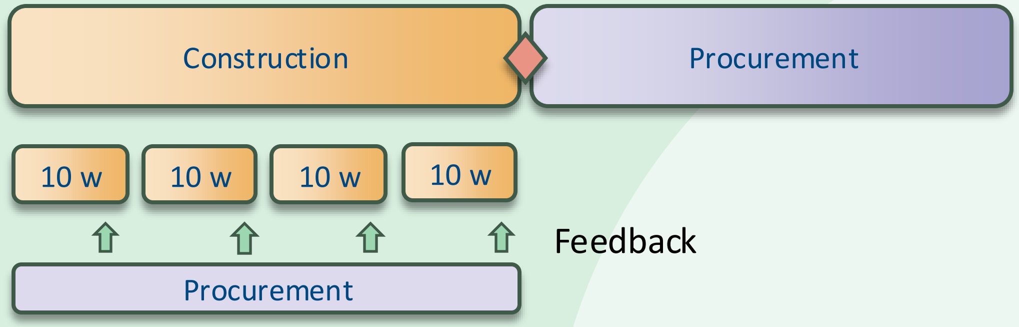
Traditional Planning

- Phase gate hand-over from construction to procurement
- Feedback from procurement forces re-design loops

Rolling wave planning

- Established 10 week iterations and invited procurement to visit
- Procurement learned about the context (requirements) instead of getting fixed specifications
- Enabled earlier feedback on first constructions avoiding late re-design

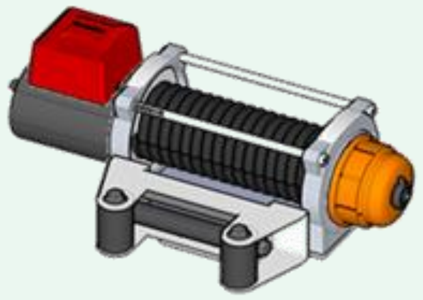


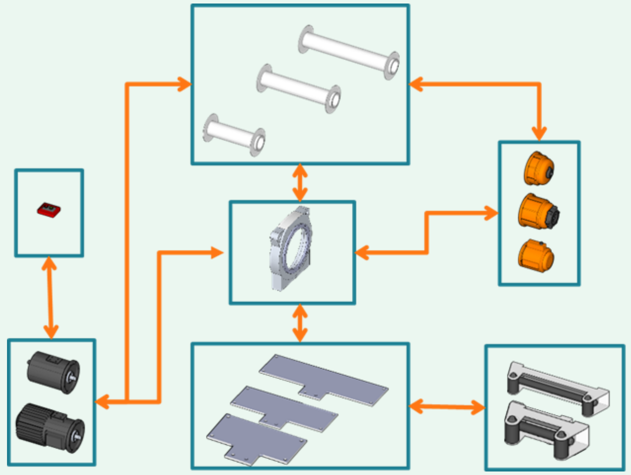


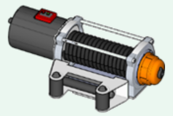
Value Based Slicing

- Ensures small learning increments

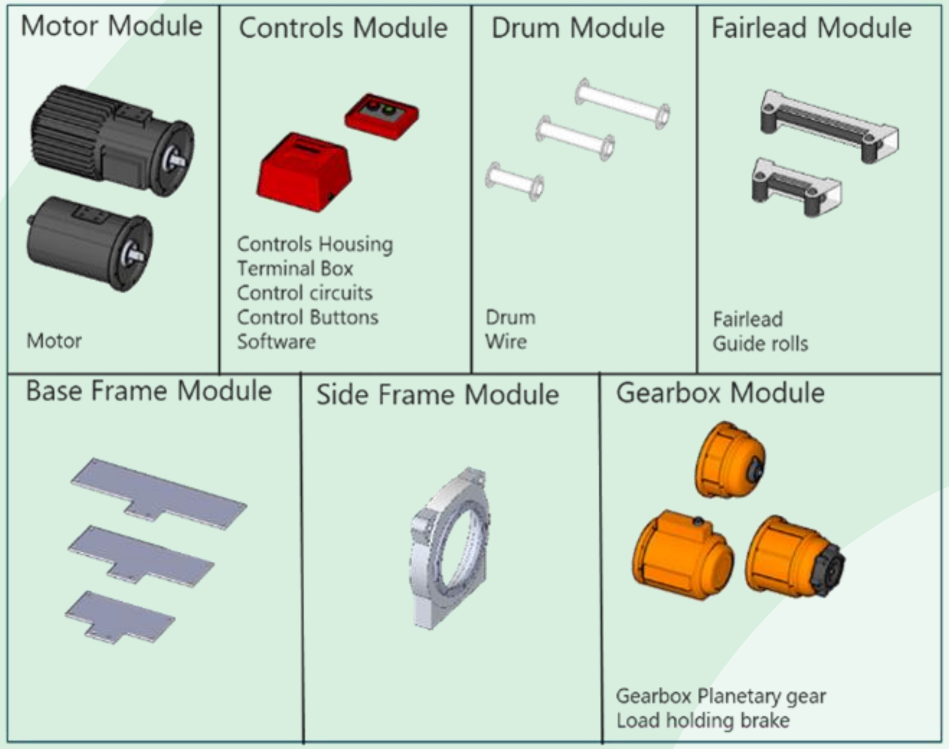
Winch Example







https://www.modularmanagement.com/



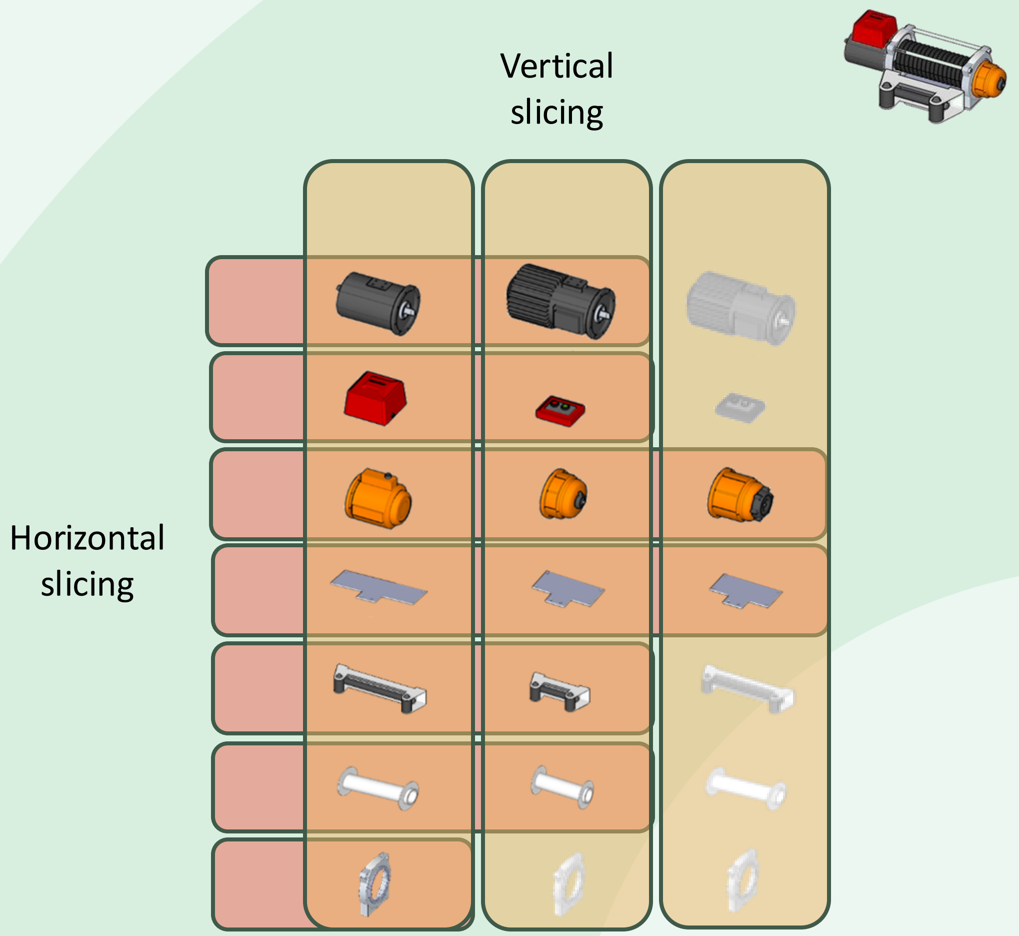
Value Based Work Packages

Horizontal: module-based

- A common way to break down a complex product is to divide it into module-based work packages
- This creates smaller work packages, but the entire system will be evaluated late
- Each part is useless until it is put together with the other modules

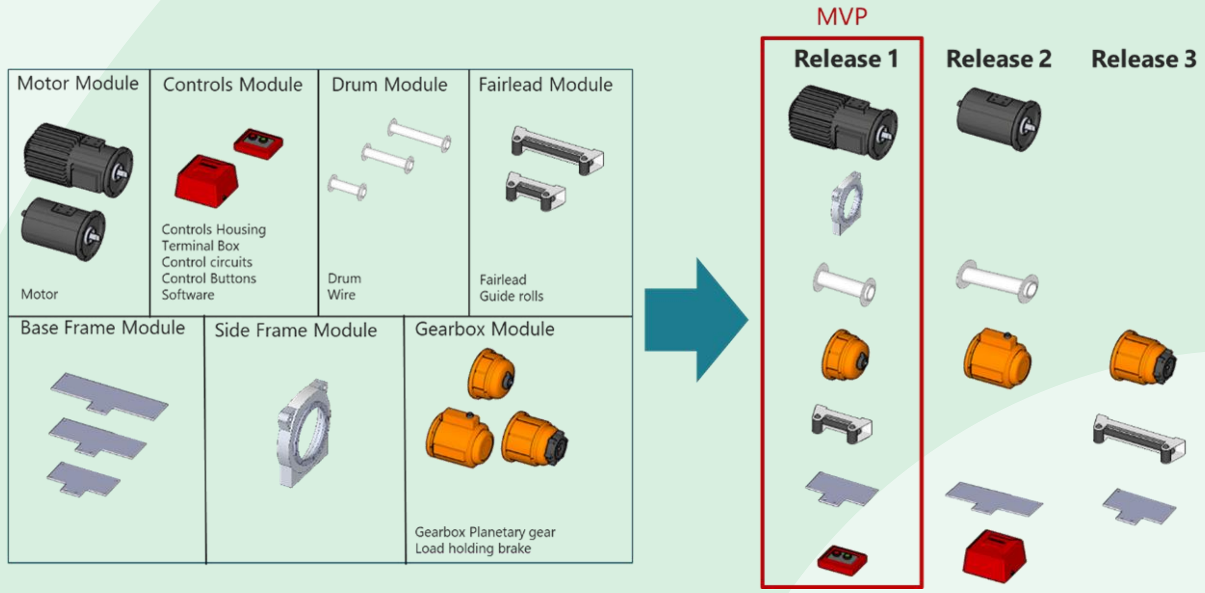
Vertical: business-value-based

- Create smaller work packages with business value vertical across the system
- More frequent integration, fast learning and early release opportunities



Minimum Viable Product (MVP)

- MVP is defined by the variants necessary to get feedback from the customers
- These variants needs to be developed first
- Other module variants can be developed later, based the learnings from the MVP
- Using the feedback a Minimum Marketable Product (MMP) can be defined



Thank you

dirk.holste@thinkflow.se

www.thinkflow.se

thinkflow