Practical information modeling

The art of taking the piece apart

Svante Ericsson
Corena
Agenda

1. Why bother about modularization – what is the S1000D idea/purpose?
2. What does S1000D suggest to do?
3. How do I go about?
Why modularize?
Why modularize?

A successful modularization will

• Support collaborative authoring
• Provide maintainable information
• Enable flexible publishing
If things go wrong ...

• Inconsistent CSDB structure
  – A certain type of item appears in several locations in the db
  – Collaboration becomes difficult
  – Difficulties to compile suitable publications
  – ...

...
If things go wrong ... 

• Ultimately, you may have to restructure your CSDB!!!
  – Unless you’re extremely lucky, this is manual work
  – Since it is manual work, it is correspondingly costly
The data module

• Constitues the smallest manageable unit in the CSDB

• For a complex system, the CSDB may eventually contain tens of thousands of data modules

A mistake may become a huge problem!!
The data module

- Information of a certain type about a certain object.
- S1000D has pre-defined some 800-900 information codes, each representing a certain type of information.
- Identified using the Data Module Code (DMC)
SNS

- Standard numbering system (SNS), the part of the DMC related to the product
- Offers a 4-level hierarchy
The goal

A breakdown of the *information about the product*

NOT a breakdown of *the product itself*
What does S1000D suggest?
The options

S1000D mentions three options

• Maintained SNS
• Example SNS
• Specify your own specialized SNS
Maintained SNS

- Generic
- Support and training equipment
- Ordnance
- General communications
- Air vehicle, engines and equipment
- Tactical missiles
- General surface vehicles
- General sea vehicles
Maintained SNS

Much used!

• Proven consistency
• Proven usability within the product domain
• Certain levels are already set
• Must be filled out with the product details
Example SNS

• Examples of what certain projects have done
• Fit for a specific purpose/target situation
• "Looks like something I’m used to, meaning I won’t have to change”
• Not maintained by the S1000D organisation!
Project specific SNS

Hopefully,

• Especially fitted for the purpose!
• Unambiguous
• Consistent
• Practical

But you are left on your own!!
How do I go about?
It is a long term engagement ...

• As a product producer you will live with your CSDB for several decades. Therefore, the CSDB must lend itself to all kinds of changes, eg use profile and environment
To identify the breakdown objects

Multi-dimensional space:

- Physical structure
- Functional structure
- Maintenance needs
- Maintenance concept
- Operational conditions
- .... !!!
To identify the objects
To identify the objects
To identify the objects
To identify the objects

- subsystem
- integration
- system
To identify the objects

Maint level X

Procedure to fix A:
1. Remove A
2. Repair A
   1. Do this
   2. Do that
   3. ...
3. Reinstall A
To identify the objects

Maint level X

Procedure to replace A:
1. Change A

Maint level Y

Procedure to repair A:
1. Do this
2. Do that
3. ...
Check questions

What happens to the CSDB if

– A new configuration is added?
– An old model is terminated?
– The product is introduced in a new environment?
– The logistics organisation is changed?
– The product gets a new/revised role?
– There is a new customer?
Check questions

In any case, changes should *only affect data modules directly concerned*!!
Conclusions
Svante’s advices

– If not impossible, start from a Maintained SNS
– Remember that all product aspects must be reflected
– Out of the box, try to anticipate future changes in usage, customer base, configurations etc and prepare for the implications
Everything should be as simple as possible, but no simpler!

Albert Einstein
Thanks for your attention!

Questions?