

# brightsight®



your  
partner  
in security  
approval



Peter van Swieten  
+ 31 15 269 2500  
[swieten@brightsight.com](mailto:swieten@brightsight.com)  
[www.brightsight.com](http://www.brightsight.com)

With assistance of  
Olaf Tettero and  
Monique Bakker

**Using tools to  
generate design  
documentation for  
CC evaluations**

## Introduction

- A report of the design tools used to create/maintain CC design documentation
  
- For different types of design tools:
  - When they can be used
  - Advantages
  - Disadvantages
  - Examples
  
- Conclusion

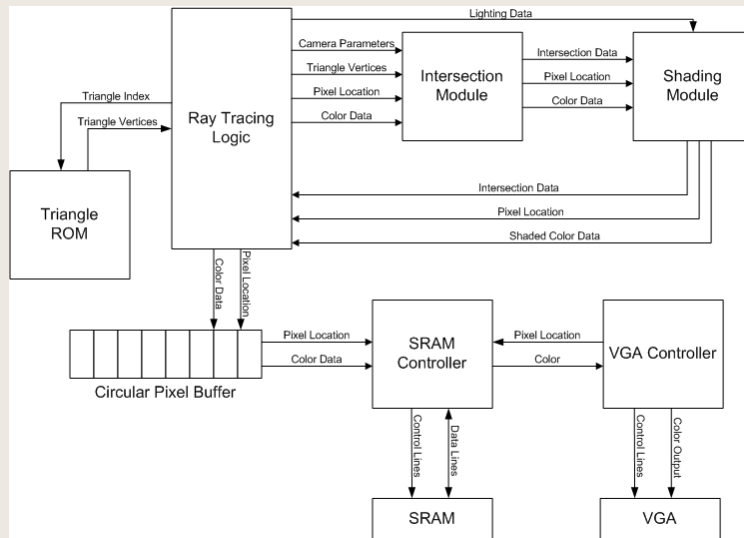
## Some necessary background information:

- What is a design?
- What are the CC requirements on such a design
- Methods to create a product and its design

# What is a design?

A design normally starts with an idea

Which results in a top-level picture:



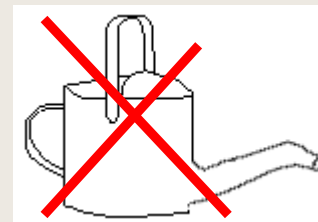
... even for a very simple product



Or you end up with something less useful:

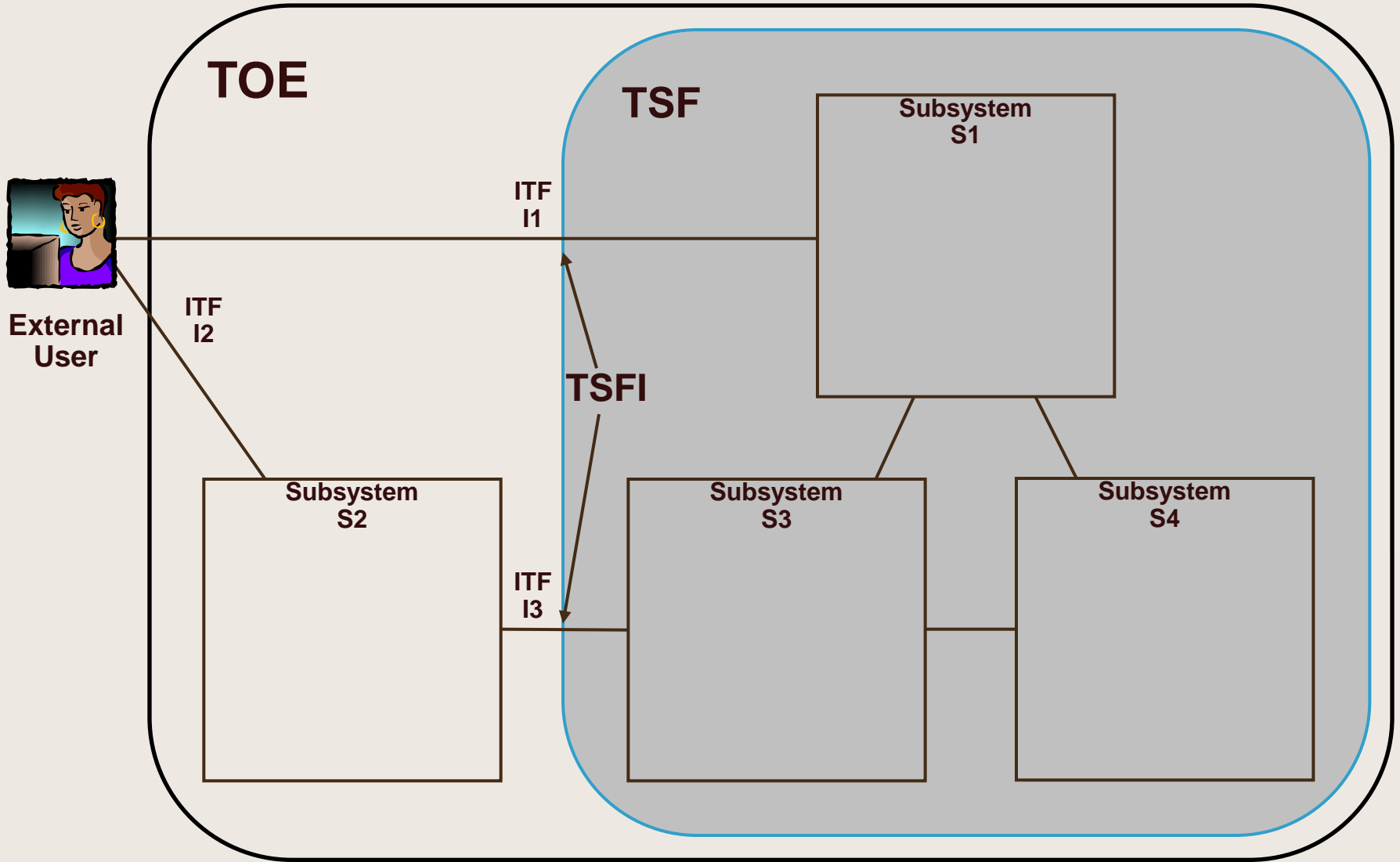


Too heavy to use

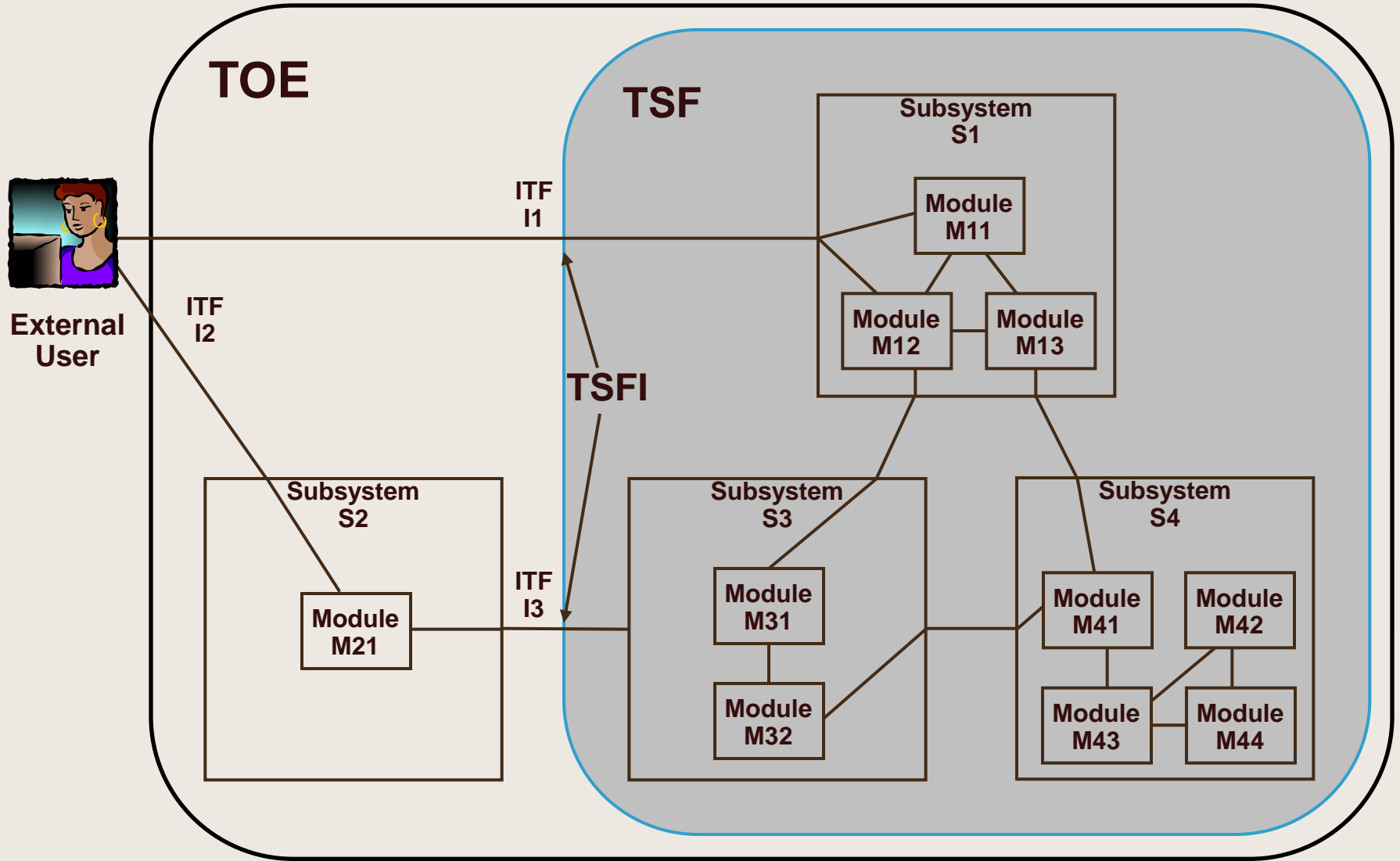


Cannot fill

# TOE design – high level



# TOE design – high level + low level



## Implementation representation

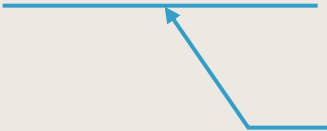
Implementation representation (According to the CC standard):

The least abstract representation of the TSF, specifically the one that is used to create the TSF itself without further design refinement.

Which means:

For a software product: The source code

For a hardware product: The source code or schematics



Out of scope for  
this presentation

## CC requirements on TOE design

For a common (EAL4) evaluation the design must include:

- Descriptions of Subsystems
- Descriptions of Modules
- Descriptions of TSFI / Interfaces

The origin of these requirements:

- > ADV\_FSP: TSFI / Interfaces
- > ADV\_TDS: Subsystems & Modules

And it has to include mappings of:

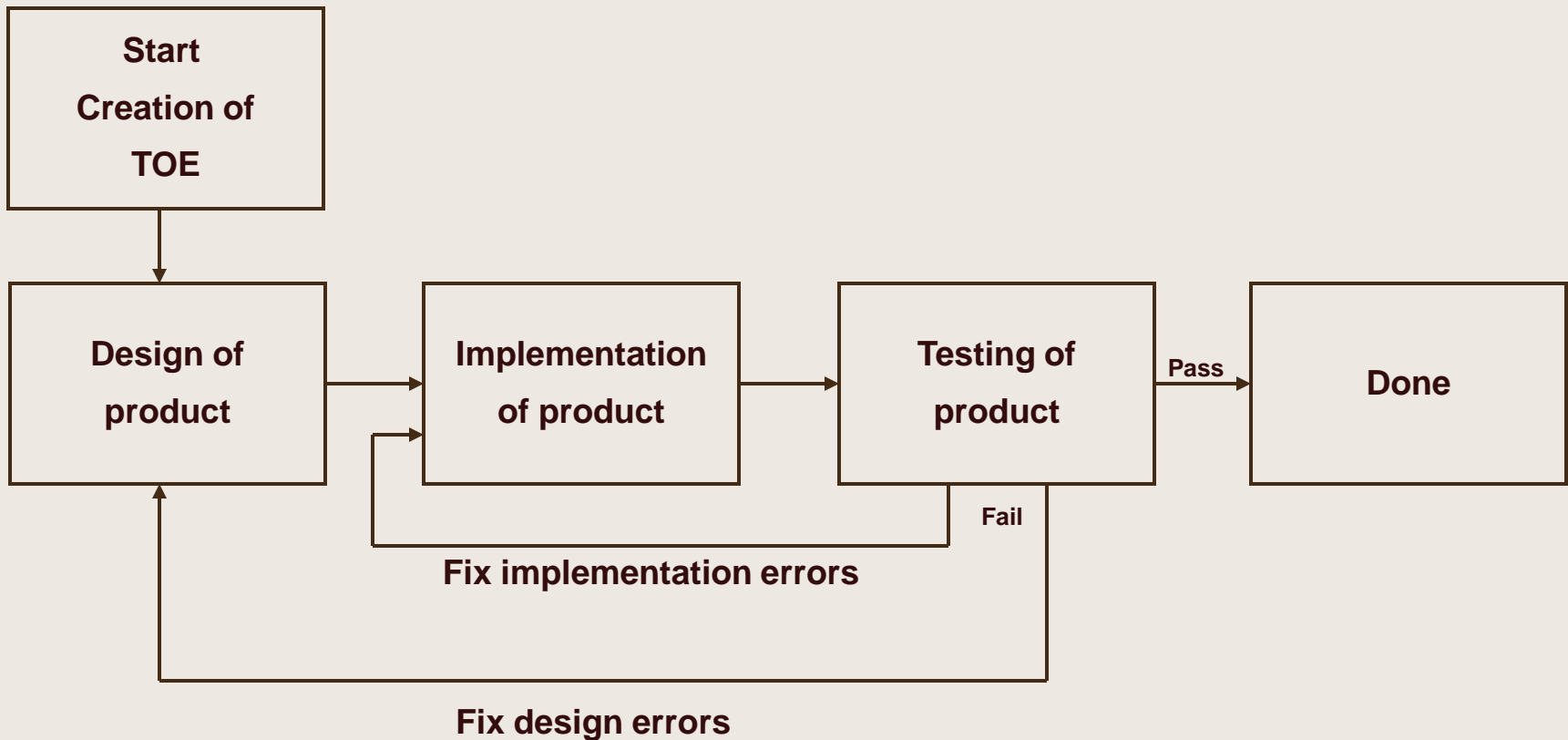
- SFRs → TSFI / interfaces
- TSFIs → modules/subsystems
- Subsystems → modules

The interesting challenges when creating/updating the TOE design are:

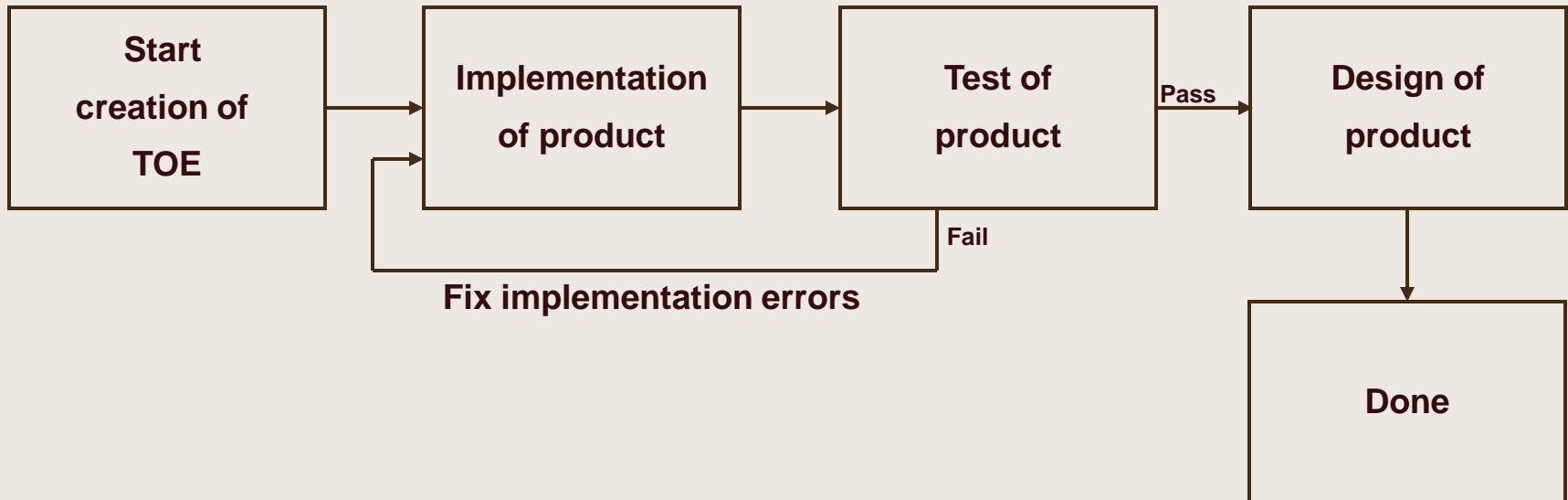
- To reach and maintain completeness
- To reach and maintain consistency

This is where the tools of  
this presentation come in

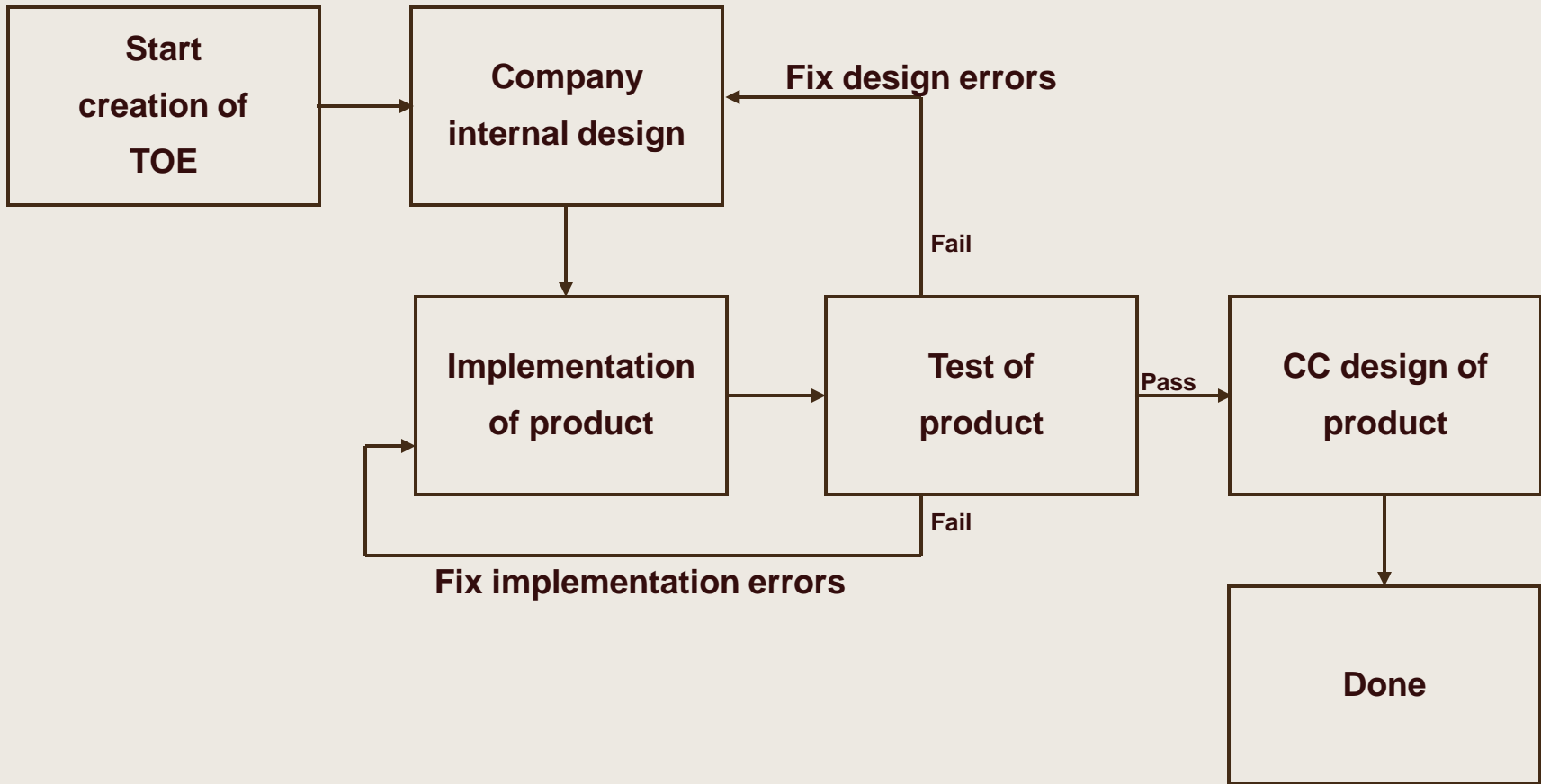
## Method to create a product and its design: Top-down approach



## Method to create a product and its design: Bottom-up approach



## Method to create a product and its design: Combined Top-down+Bottom-up approach



## Design tools

The different categories of design tools:

- Interactive modeling tool for TOE design that creates (and updates in case of design changes) the implementation representation.
- Design documentation generators
- Word processor and drawing tools

Tools addressed in terms of:

- With which design method(s) they can be used
- Advantages
- Disadvantages

Give an example

## The most advanced: Interactive modeling tool for TOE design(1)

Application:

- Top-down approach only

Examples:

Software: IBM Rational Rhapsody,  
Enterprise Architect

Hardware: HDL designer

Advantages:

- Design checked by tool, implementation representation managed by tool
  - Consistency and completeness guaranteed by tool. No checks required by evaluator.
  - Design is easily accessible (browseable). Subsystem-module mapping implicit.
  - The tool saves a lot of time:
    - The tool generates the framework of the source code.
    - In case of design changes: The tool updates the source code to reflect the design.
      - Enables the programmer to focus on producing code.
- Full control over presentation (i.e. format) of design.
- For some tools: Possible to use with an existing implementation representation (re-use) through reverse engineering.

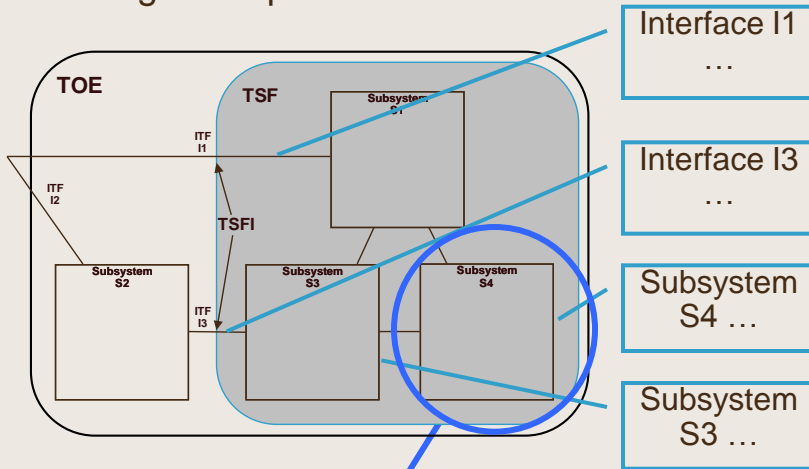
## The most advanced: Interactive modeling tool for TOE design(2)

Disadvantages:

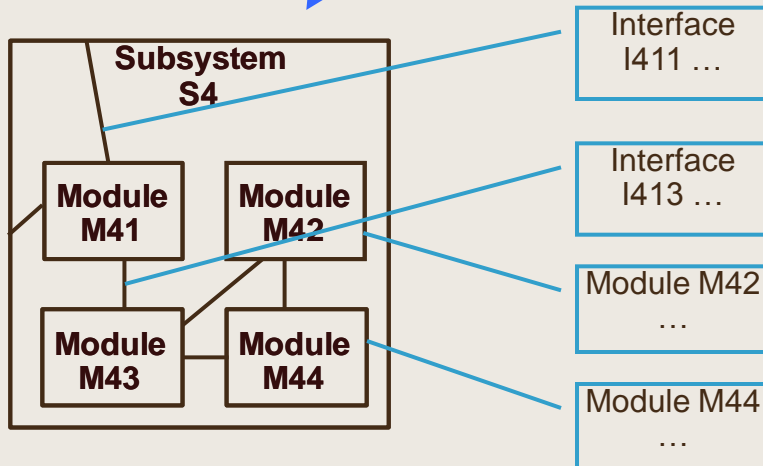
- Works for a limited set of implementation representation languages only.
- Design documentation is not sequential. Therefore reading a printout is not useful.
- Tool not designed for CC:
  - The designer remains responsible to include certain CC specific information.

## The most advanced: Interactive modeling tool for TOE design(3)

Step 1: Create the high level design including descriptions



Step 2: Add the low level design details including descriptions



Step 3: Have the tool check the design and update the design (Step 1 & 2) as appropriate.

Step 4: Have the tool create/update the implementation representation. This results in a tree with source files.

Step 5: Include the functionality (i.e. actual code) into the source files at the appropriate locations.

Step 6: build the TOE

## Design documentation generator (1)

### Application:

- Any design approach

### Examples:

Software: Doxygen, Javadoc  
Hardware: Doxygen

### Advantages:

- Design generated from implementation representation
  - Consistency and completeness with implementation representation guaranteed. No checks required by evaluator.
  - Design is easily accessible (browseable). Subsystem-module mapping implicit.
  - The tool saves time:
    - In case of implementation changes: The tool updates the design documentation.
- Possible to use with an existing implementation representation (re-use).

## Design documentation generator (2)

### Disadvantages:

- Works for a limited set of implementation representation languages only.
- Design documentation is not sequential. Therefore reading a printout is not useful.
- The designer has to format the implementation representation in a specific format for the tool to work properly.
- Tool not designed for CC:
  - The designer remains responsible to include certain CC specific information.

### Usage:

Step 1: Embed the documentation of the design into the implementation representation

Step 2: Run the tool to generate the documentation from the implementation representation

➔ Output documentation: A tree of browseable documents (with e.g. firefox).

## The most used: Word processor and drawing tools

### Application:

- Any design approach

### Examples:

Microsoft office, OpenOffice

### Advantages:

- Can easily satisfy any CC requirement because of full control over the text order, format and the possibility to embed pictures.
- Can be printed and read sequentially
- Can be used with any type/language of TOE implementation representation

### Disadvantages:

- Prone to completeness problems.
- Prone to consistency problems.
- No automated relation between implementation representation and design which causes even more completeness and consistency problems in case of iterative steps in the design process.

## Which tool type to use?

- First choice:  
Interactive modeling tool for TOE design.
- If not possible to use first choice, revert to second choice:  
Design documentation generator.
- If not possible to use second choice, revert to:  
Word processor and drawing tools.
  
- But: For a very small TOE (e.g. a data diode) use:  
Word processor and drawing tools.

## Conclusion

- There are various tools for the creation of CC design documentation:
  - Interactive modeling tool for TOE design – The most advanced
  - Design documentation generator
  - Word processor and drawing tools – The most used
  
- The most advanced tool helps best to create and maintain complete and consistent design documentation
  
- The most used tool does not help to create and maintain complete and consistent design documentation
  
- Sometimes (depending on the TOE implementation representation language and its design process) one has to revert to the word processor and drawing tools
  
- For a very small TOE the word processor and drawing tools are still the preferred tool.

## Contact information

Bright sight BV  
Delftechpark 1  
2628 XJ Delft  
The Netherlands



Peter van Swieten  
+ 31 15 269 2500  
[swieten@brightsight.com](mailto:swieten@brightsight.com)  
[www.brightsight.com](http://www.brightsight.com)