

# Mixing Re-Use and Model-Based Development The CHEOPS Payload SW Experience

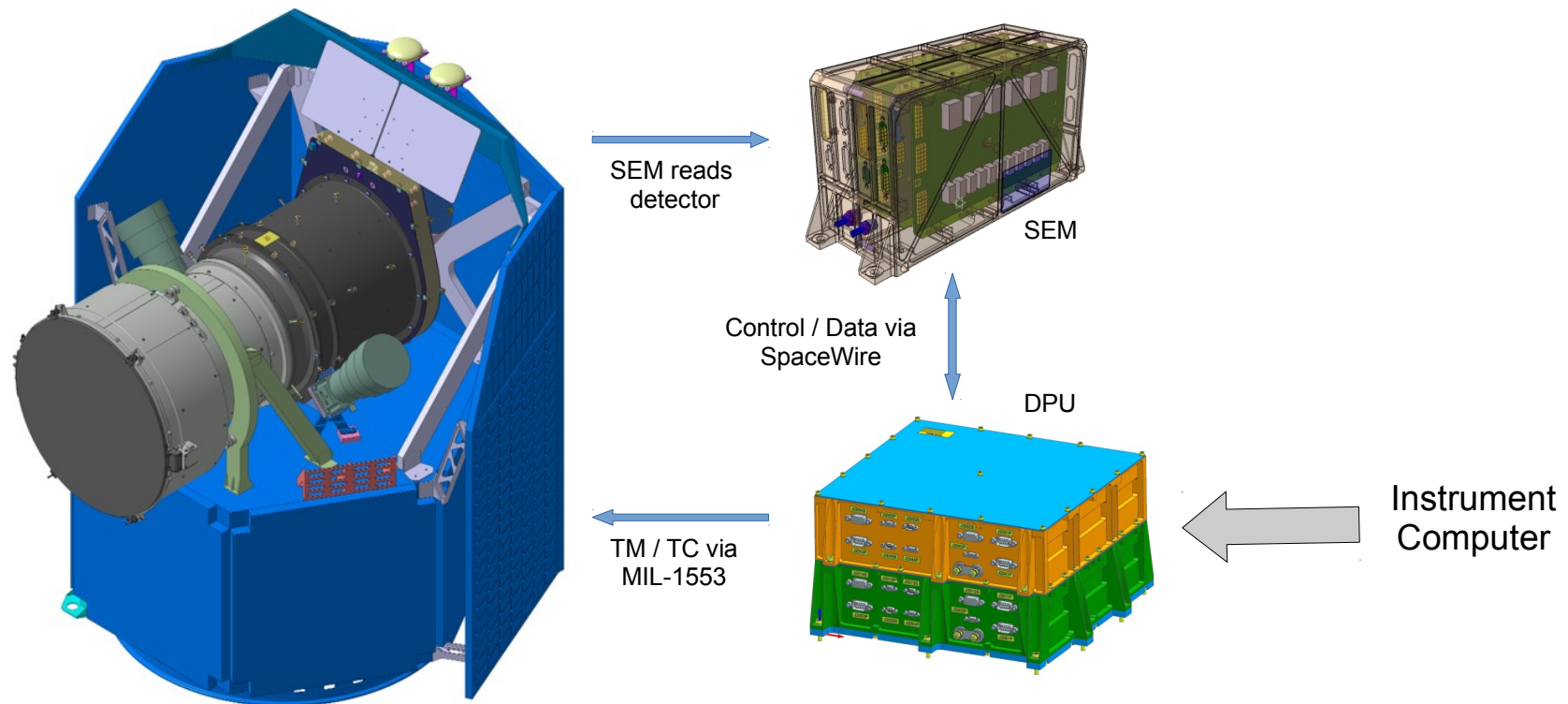
A. Pasetti<sup>1</sup> & M. Opprecht<sup>1</sup>

<sup>1</sup>*P&P Software GmbH, <http://www.pnp-software.com>*

# The CHEOPS Instrument



- **Characterizing ExOPlanet Satellite (CHEOPS):** single-instrument spacecraft to be launched in 2018 to study exo-planets



- Instrument Prime Contractor: **University of Bern**
- Instrument Software Development: **University of Vienna**

# CHEOPS Payload Software

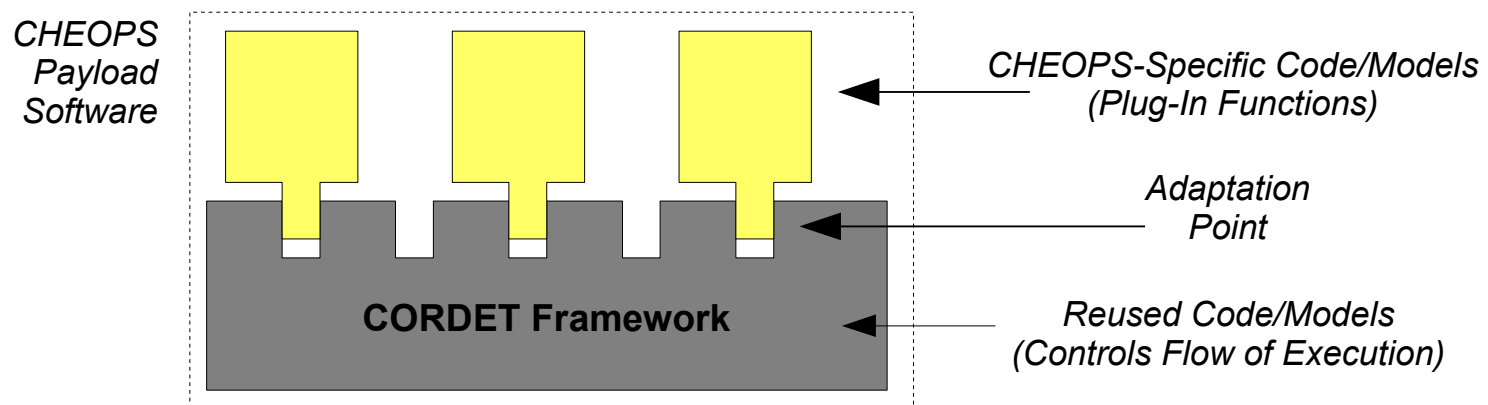


- Main technical features:

- **Provider of PUS Services** to Spacecraft Computer (17 Services)
- **User of PUS Service** from Sensor Electronics Module (7 Services)
- **Re-routing** of telecommands and telemetry reports
- Management of **16 Gbytes of Flash** Memory Storage
- Implementation of **On-Board Science Data Processing**
- **Dual-Core Architecture** (LEON3FT)

- Development Approach:

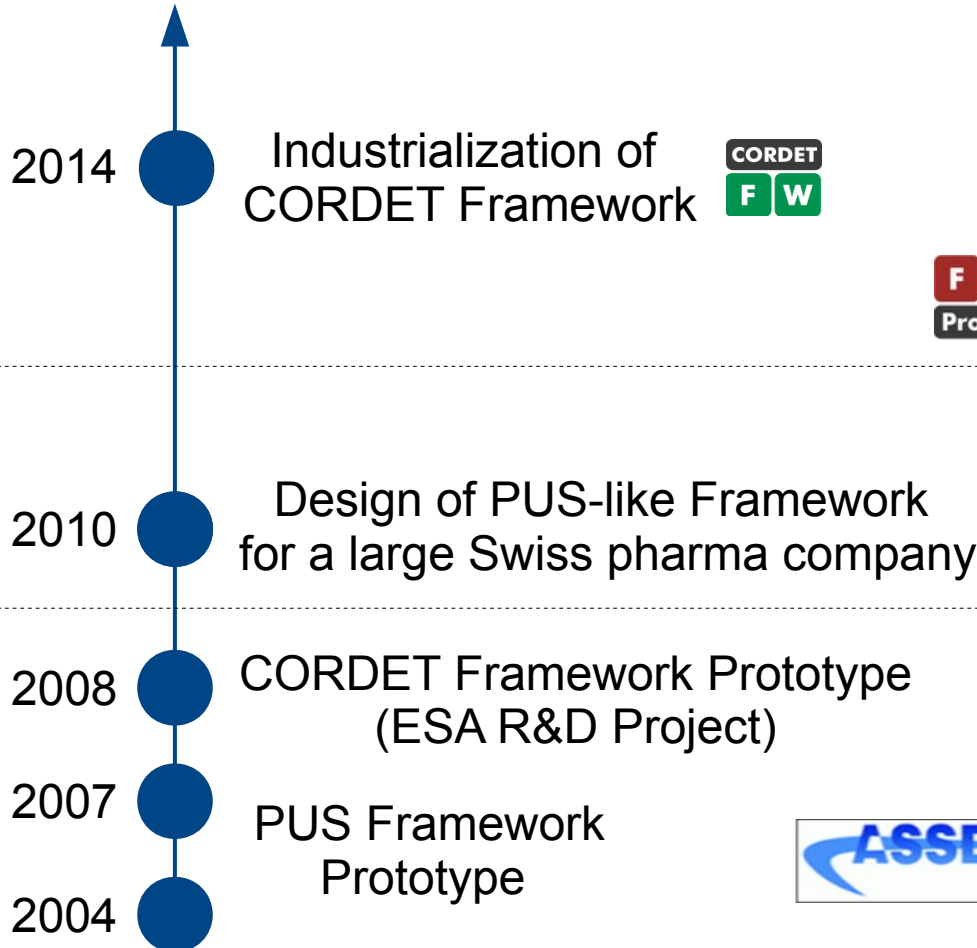
- Development as an instantiation of the **CORDET Framework** -► Reuse + MBSSE
  - CORDET Framework is defined both at model and code level
- Framework customization developed as **FW Profile Models** -► MBSSE



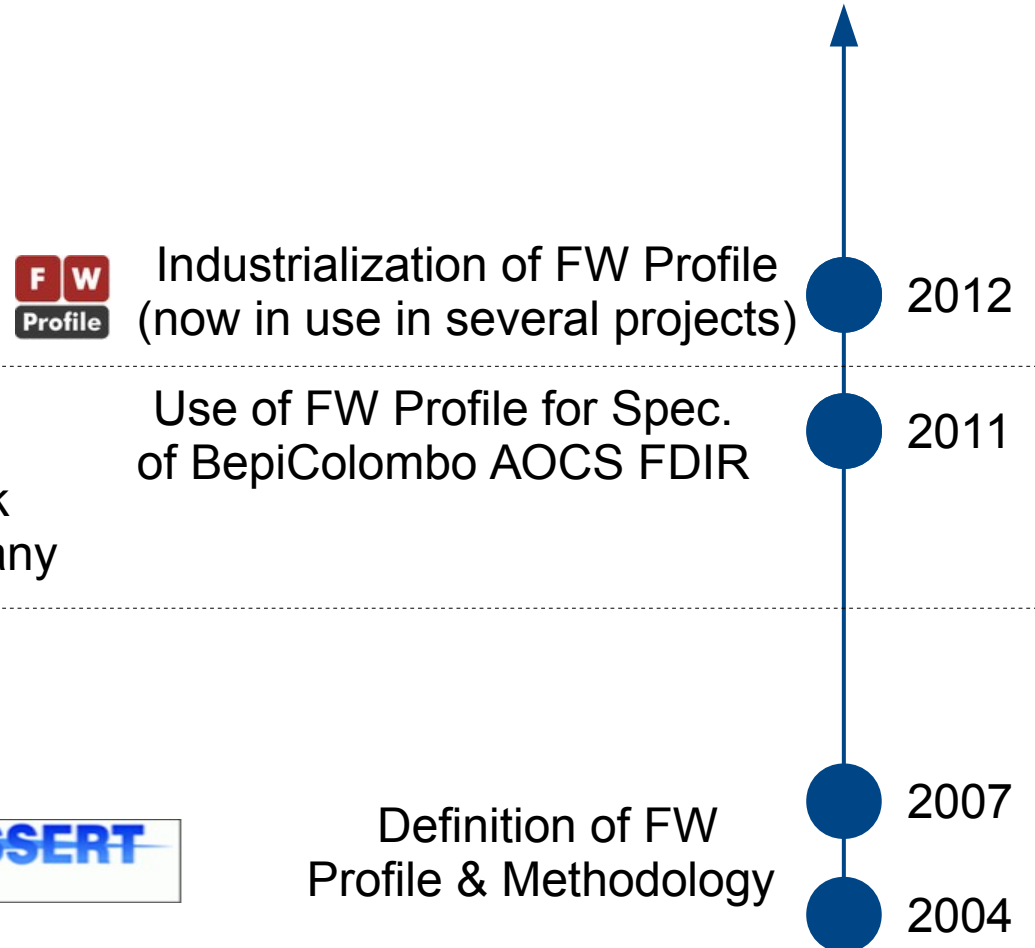
# History & Heritage



## CORDET Framework



## FW Profile



# What is the FW Profile?



- The **FW Profile** is a UML profile which consists of:
  - Specification of four **basic concepts** to model application behaviour
    - **State Machines** to model state-dependent functional behaviour
    - **Procedures** (Activity Diagrams) to model sequential functional behaviour
    - **Real-Time Containers** to model timing behaviour
    - **Adaptation Points** to support definition of reusable components
  - **C-language implementation** of these concepts to support translation of specification models into code
  - **Qualification Data Package** for the C implementation to support certification of end-applications
  - **Web-based tool** to build profile-compliant models and generate their code
  - **Free/open licencing** model (LGPL): [www.pnp-software.com/fwprofile](http://www.pnp-software.com/fwprofile)

# What is the CORDET Framework?



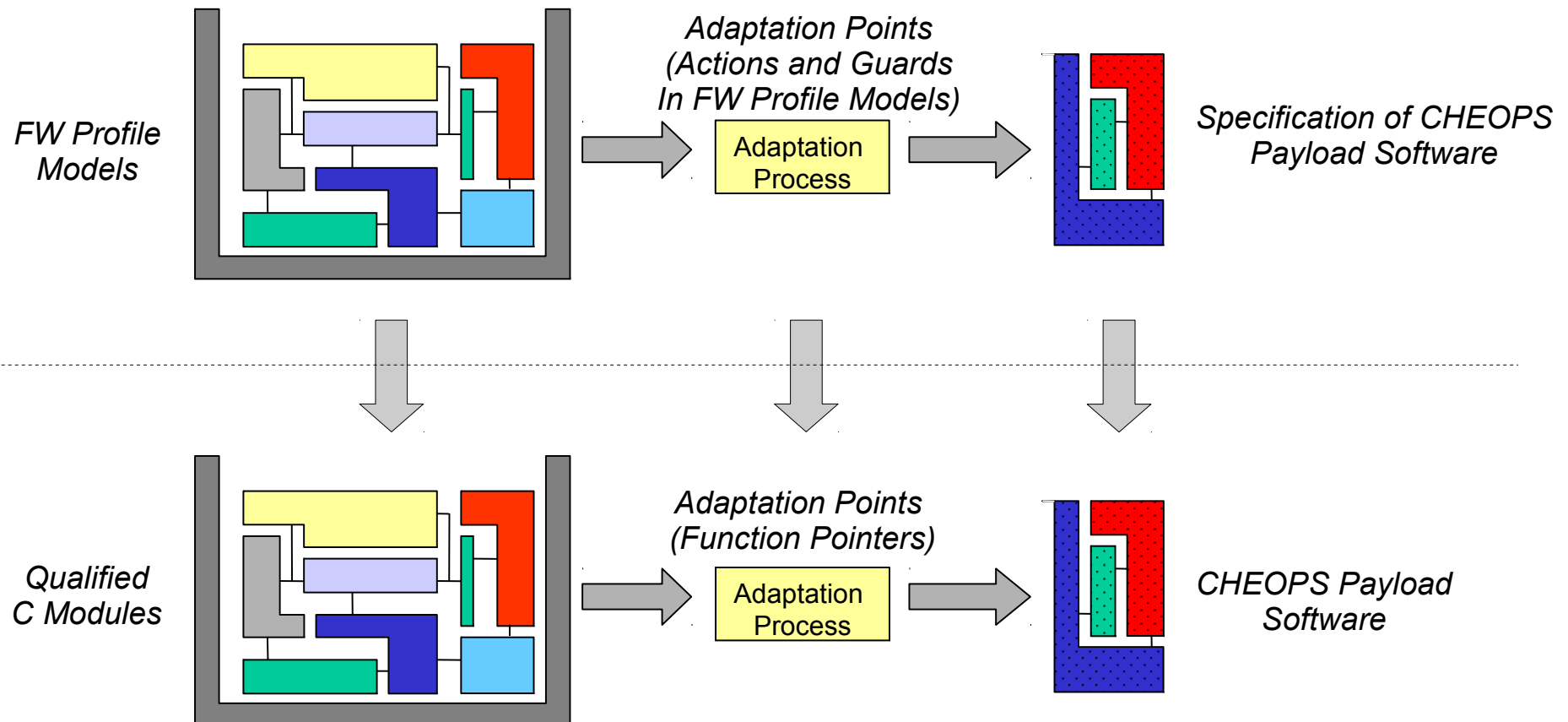
- The **CORDET Framework** consists of:
  - Set of **FW Profile models** which specify:
    - A generic architecture to manage incoming and out-going **commands**
    - A generic architecture to manage incoming and out-going **reports**
    - The **adaptation points** where application specific behaviour can be plugged in
  - **C-language implementation** of these models to support translation of specification models into code
  - **Qualification Data Package** for the C implementation to support certification of end-applications
  - **Free/open licencing** model (LGPL): [www.pnp-software.com/cordetfw](http://www.pnp-software.com/cordetfw)
  - **Web-Based Tool** to automate the framework-instantiation process

*Currently Under Development*

A thin black arrow originates from the text 'Currently Under Development' and points diagonally upwards and to the right, ending at the 'Web-Based Tool' bullet point in the list above.

# A Model-Based Framework

- The CORDET Framework exists both at **model level** and at **code level**
  - The instantiation process is done at both levels
  - The specification is created from the models and the implementation from the code



# Impact on Design Process



- Requirements for the CHEOPS Instrument Software have been written using the abstractions provided by the CORDET Framework
  - Requirement definition effectively consisted in defining the close-out for each framework adaptation point
  - **Requirement Baseline serves also as Technical Specification** and is used as input for software design process
- Architectural Design Process effectively disappears
- CHEOPS Instrument Software Development
  - First release of Application SW Technical Specification: 1 month
  - First release of Architectural Design Document: < 1 month



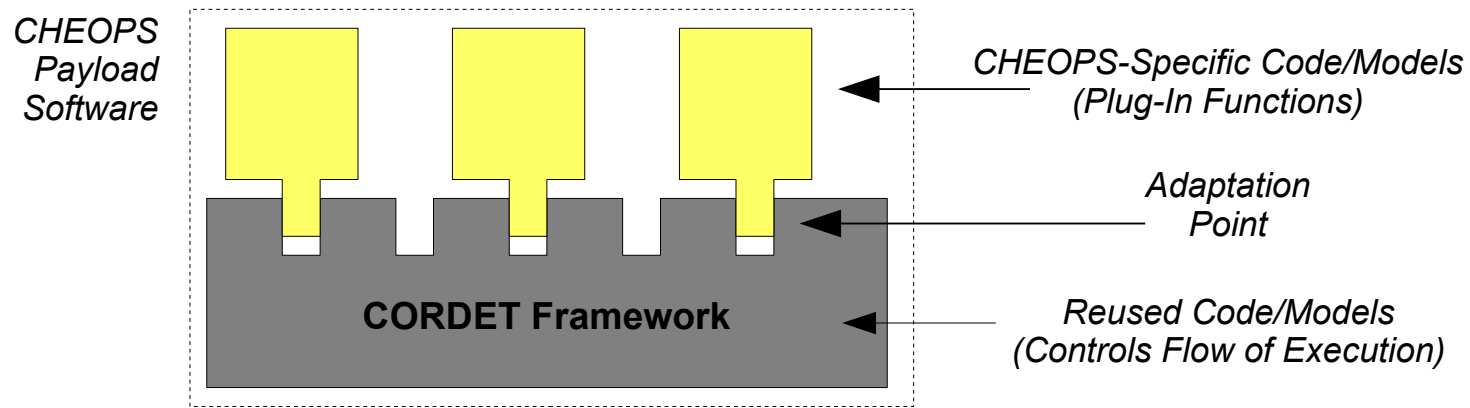
# Impact on Qualification Process



- The CORDET FW is provided with a **Qualification Data Package**:
  - Formal specification of framework behaviour through **UML Models** and **Requirements**
    - **Traceability** to code
    - **Traceability** to verification evidence
  - **Test Suite** with 100% code, branch, and condition coverage (excluding system call error branches)
  - **Doxygen Documentation** for the entire code base
  - **User Manual & Demo Application**
- The selected adaptation mechanisms require **no change to the source code** of the framework code
  
- The qualification data package of the framework is **imported** into the qualification data package of the CHEOPS Instrument Software
  - ➔ **No need for re-qualification** of code imported from the framework

# Application Software Structure

- The Framework acts as a **domain-specific operating system**
- Application-specific code consists of **call-back functions** which are registered with the framework and are called by the framework
- A model-based process for application-specific code was used:
  - The specification consisted of FW Profile Models
  - The state machine and activity diagram code was generated automatically
  - Manual development was restricted to code implementing individual actions and guards
  - The manually-developed code was mostly “linear”



# Tool Support - FW Profile Editor



[www.pnp-software.com/fwprofile/editor-5.00](http://www.pnp-software.com/fwprofile/editor-5.00)

- Files
- Code
- Undo
- To Self
- Delete
- Initial
- Final
- Decision
- Action
- Note
- Help

FW Profile

(c) P&P Software  
version 5.00

The procedure transfers a number of blocks from a RAM Data Area to a Target Flash-Based File (FBF). The address of the RAM Data Area, the identifier of the FBF and the number of blocks to be transferred are procedure arguments

Flag\_1 is true if the FBF is disabled, or invalid, or if the spacecraft is crossing the SAA (i.e. isSaaActive is true)

```
graph TD
    Start(( )) --> D1{ }
    D1 -- "[ Flag_1 ]" --> N5[N5: Load EVT_FBF_SAVE_DENIED with FBF identifier as parameter]
    D1 -- "[ ! Flag_1 ]" --> N1[N1: Call IBSW operation to open the Target FBF]
    N5 --> D2{ }
    N1 --> N2[N2: NOP]
    N2 -- "[ Wait FBF_BLK_WR_DUR Cycles ]" --> D3{ }
    D3 -- "[ (All requested blocks have been written) || Flag_1 ]" --> D2
    D3 -- "[ (Not all requested blocks have been written) && !Flag_1 ]" --> N3[N3: Call IBSW operation to transfer the next block from RAM Data Area to the Target FBF]
    N3 -- "[ Wait FBF_BLK_WR_DUR Cycles ]" --> D3
    D2 -- "[ ! Flag_1 ]" --> N4[N4: Call IBSW operation to close the Target FBF]
    D2 -- "[ Flag_1 ]" --> N6[N6: Load EVT_FBF_SAVE_ABORT with FBF identifier and number of blocks written as parameters]
    N4 --> End(( ))
    N6 --> End
```

### Global Properties

Procedure Name:

Project name:

User-Defined global variables:  
  =  +

Custom includes:

Notes:

Show control flow order: ?  
 no  yes

Show text on diagram:  
 auto  functions/code  descriptions

# Contentious Statements



- If we use a model-based approach to software development, then **requirements baseline** and **technical specification** will (tend to) merge
- If we want to design reusable software, then we need to explicitly **model adaptation mechanisms**
- Code generated from models should “look like” manually developed code so that we can **qualify the generated code** rather than the generator