Design document

A design document defines the data structure to be exchanged with the controllers. A design document is basically an XML document containing a set of ordered blocks of registers.

This document is composed of the following nodes:

- **Information node.** This node contains the following sub-nodes:
  - **Owner** is the one who created the document. He has write access on the design document and can deploy the corresponding class. Owner only can add Editors and Users on the design.
  - **Editor** (optional) has write access on the design document and can deploy the corresponding class. He cannot add Editors or Users on the class.
  - **User** (optional) has read only access on the Design document but can deploy the corresponding class. He cannot add Editors or Users on the class.
- **SILECS-Class** defines a clear interface between the controller process and the client software for the control of a particular equipment system. A Silecs-Class is uniquely identified by his name and version. It provides a list of blocks and registers to be exchanged with the controllers and defines their accessing policies. A Silecs class is composed of a list of Blocks.
- **Block** is a group of registers that match the same functionality and share the same access mode (read-only,write-only, read-write). This type of memory organization optimizes the communication transfers and fits with the standard Device/Property model of the BE/CO controls. The block name must be unique in the scope of the Silecs-class. Each data transaction that is done from the client side (send/recv) relies on the blockname. The list of blocks inside the Silecs-class will occupy contiguous segments of PLC memory following their exact ordering. A block contains one or more registers.
- **Register** is the base entity of the Silecs data model.The register name must be unique in the scope of the Silecs-class (therefore same name cannot be reused within different blocks). Different data types are supported. The following table shows how the different Silecs data types are converted during the controller code generation.

SILECS DATA TYPE	SIEMENS PLC	SCHNEIDER PLC	BECKHOFF PLC	NATIONAL INSTRUMENTS	RABBIT MICROCONTROLLER	NOTES
uint8	BYTE	BYTE	BYTE	U8	uint8_t	
int8	CHAR	WORD	SINT	I8	int8_t	
uint16	WORD	WORD	WORD	U16	uint16_t	
int16	INT	INT	INT	I16	int16_t	
uint32	DWORD	DWORD	DWORD	U32	uint32_t	
int32	DINT	DINT	DINT	I32	int32_t	
uint64	-	-	-	U64	uint64_t	
int64	-	-	-	I64	int64_t	
float32	REAL	REAL	REAL	SGL	Float	
float64	-	-	LREAL	DBL	-	
date	DT	DT	DT	DBL (*)	dt (**)	
char	CHAR	WORD	SINT	U16	uint16_t	DEPRECATED
byte	BYTE	BYTE	BYTE	I16	int16_t	DEPRECATED
word	WORD	WORD	WORD			DEPRECATED
dword	DWORD	DWORD	DWORD			DEPRECATED
int	INT	INT	INT	U32	uint32_t	DEPRECATED
dint	DINT	DINT	DINT	I32	int32_t	DEPRECATED
real	REAL	REAL	REAL	SGL	float	DEPRECATED
dt	DT	DT	DT	DBL (*)	dt (**)	DEPRECATED
String	STRING	STRING	STRING	STRING	char[]	

(*) Same space as a date is reserved. It actually transfers double. No way of encoding the date into the double has been designed yet.

(**) Custom data structure automatically generated. It implements on Rabbit microcontroller the same data type used for SCHNEIDER PLCs.

By default a register is declared as scalar (array-dim1="1"). Changing the value of array-dim1 to a value > 1 the register become an array. Bidimentional array can be created by adding the optional attribute array-dim2.The list of registers inside the Blocks will occupy contiguous segments of controller memory following their exact ordering.

A register can be retentive: this is a constant or persistent data which require an initial-value and which is retained in case of controls reset (e.g. persistent settings, configuration parameters, etc.). The controller expert can choose to provide the initialization value from the controller or from the FEC:

- **MASTER**: The data is retentive and its initial-value is stored in the controller memory.
- **SLAVE**: The data is retentive and its initial-value is provided by the client process.
- **NONE**: The register does not require any synchronization (volatile data).