



# Using document templates to structure inputs and autogenerate database

[tilen.zagar@cosylab.com](mailto:tilen.zagar@cosylab.com)

# Challenges of developing (control system) software?

## PIVOTING

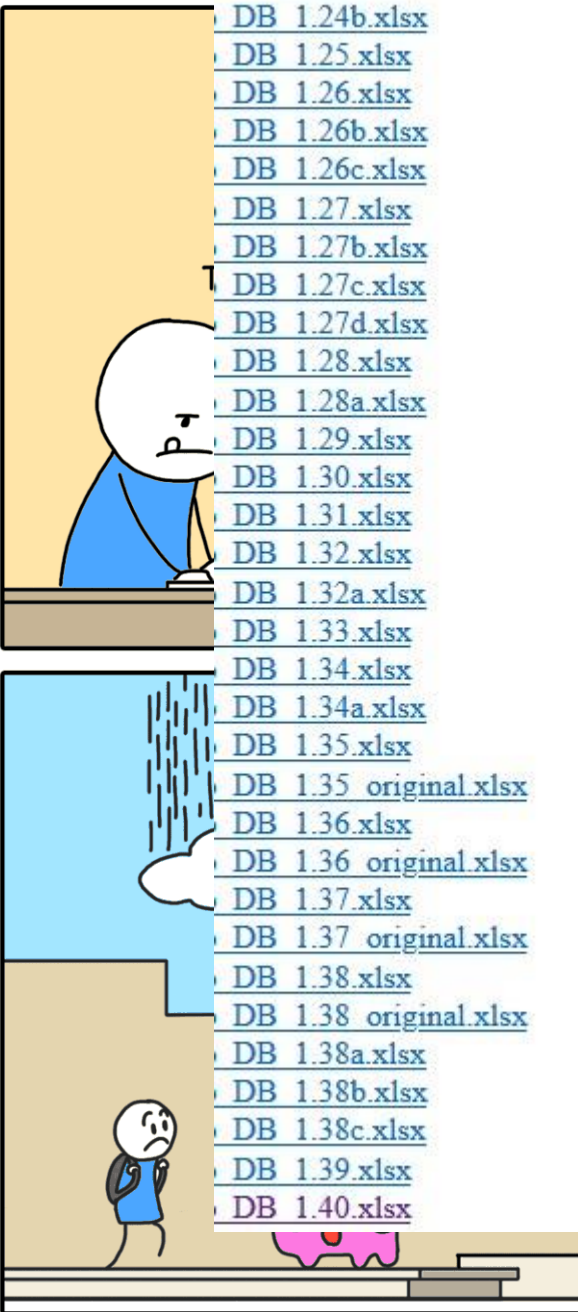


- [PLC IOC exchange 0.1.xlsx](#)
- [PLC IOC exchange Comments.xlsx](#)
- [PLC IOC exchange Examples.xlsx](#)
- [PLC IOC exchange V0 03.xls](#)
- [PLC IOC exchange V0 04.xls](#)
- [PLC IOC exchange V0 05.xls](#)
- [PLC IOC exchange V0 07.xls](#)
- [PLC IOC exchange V0 08.xls](#)
- [PLC IOC exchange V0 09.xls](#)
- [PLC IOC exchange V0 10.xls](#)
- [PLC IOC exchange V0 11.xls](#)
- [PLC IOC exchange V0 12.xls](#)
- [PLC IOC exchange V0 13.xls](#)
- [PLC IOC exchange V0 14.xls](#)
- [PLC IOC exchange V0 15.xls](#)
- [PLC IOC exchange V0 16.xls](#)
- [PLC IOC exchange V0 17.xls](#)
- [PLC IOC exchange V0 18.xls](#)
- [PLC IOC exchange V0 19.xls](#)
- [PLC IOC exchange V0 20.xls](#)
- [PLC IOC exchange V0 21.xls](#)
- [PLC IOC exchange V0 22.xls](#)
- [PLC IOC exchange V0 23.xls](#)
- [PLC IOC exchange V0 24.xls](#)
- [PLC IOC exchange V0 25.xls](#)
- [PLC IOC exchange V0 26.xls](#)
- [PLC IOC exchange V0 27.xls](#)
- [PLC IOC exchange V0 28.xls](#)
- [PLC IOC exchange V0 29.xls](#)
- [PLC IOC exchange V0 30.xls](#)
- [PLC IOC exchange V0 31.xls](#)
- [PLC IOC exchange V0 32.xls](#)
- [PLC IOC exchange V0 33.xls](#)
- [PLC IOC exchange V0 34.xls](#)
- [PLC IOC exchange V0 35.xls](#)
- [PLC IOC exchange V0 36.xls](#)
- [PLC IOC exchange V0 37.xls](#)
- [PLC IOC exchange V0 38.xls](#)
- [PLC IOC exchange V0 39.xls](#)
- [PLC IOC exchange V0 40.xls](#)

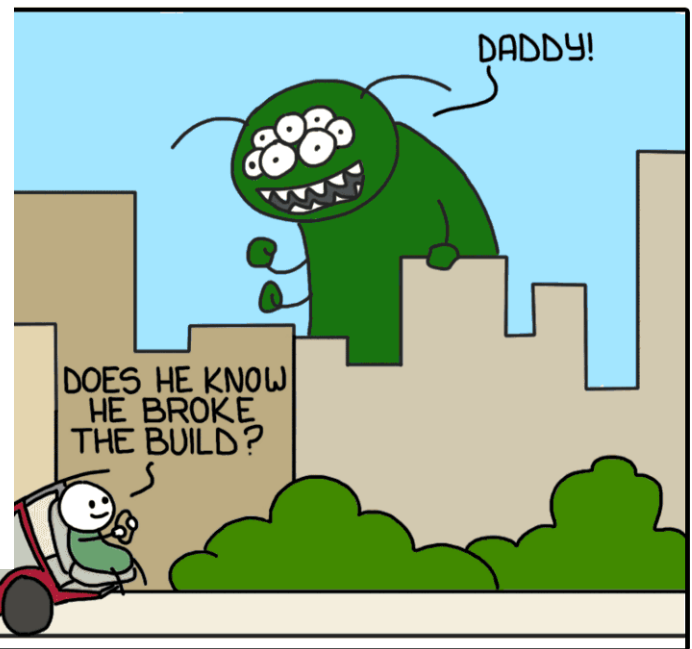
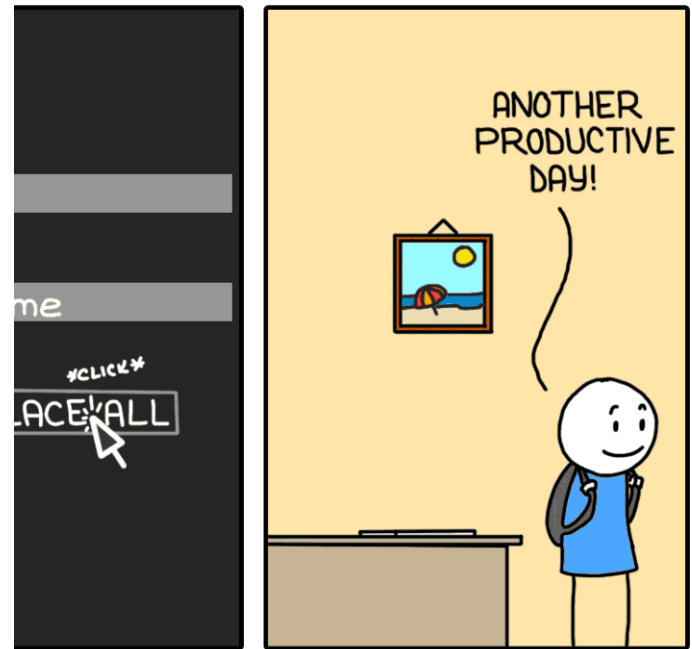
- [DB.xlsx](#)
- [DB 0.3 Lkt-1.xlsx](#)
- [DB 0.5.xlsx](#)
- [DB 0.6.xlsx](#)
- [DB 0.62.xlsx](#)
- [DB 0.71.xlsx](#)
- [DB 0.8.xlsx](#)
- [DB 0.81.xlsx](#)
- [DB 0.82.xlsx](#)
- [DB 0.83.xlsx](#)
- [DB 0.84.xlsx](#)
- [DB 0.86.xlsx](#)
- [DB 0.87.xlsx](#)
- [DB 0.88.xlsx](#)
- [DB 0.88b.xlsx](#)
- [DB 0.89.xlsx](#)
- [DB 0.90.xlsx](#)
- [DB 0.90b.xlsx](#)
- [DB 0.90c.xlsx](#)
- [DB 0.91.xlsx](#)
- [DB 0.92.xlsx](#)
- [DB 0.93.xlsx](#)
- [DB 0.94.xlsx](#)
- [DB 0.95.xlsx](#)
- [DB 0.95a.xlsx](#)
- [DB 0.95b.xlsx](#)
- [DB 0.95c.xlsx](#)
- [DB 0.95d.xlsx](#)
- [DB 0.96.xlsx](#)
- [DB 0.97.xlsx](#)
- [DB 0.98.xlsx](#)
- [DB 0.98a.xlsx](#)
- [DB 0.99.xlsx](#)
- [DB 1.00.xlsx](#)
- [DB 1.1.xlsx](#)
- [DB 1.2.xlsx](#)
- [DB 1.21.xlsx](#)
- [DB 1.22.xlsx](#)
- [DB 1.22A.xlsx](#)
- [DB 1.22B.xlsx](#)
- [DB 1.23.xlsx](#)
- [DB 1.24.xlsx](#)

KEYUSER

## REPLACE ALL



- [DB 1.24.xlsx](#)
- [DB 1.24b.xlsx](#)
- [DB 1.25.xlsx](#)
- [DB 1.26.xlsx](#)
- [DB 1.26b.xlsx](#)
- [DB 1.26c.xlsx](#)
- [DB 1.27.xlsx](#)
- [DB 1.27b.xlsx](#)
- [DB 1.27c.xlsx](#)
- [DB 1.27d.xlsx](#)
- [DB 1.28.xlsx](#)
- [DB 1.28a.xlsx](#)
- [DB 1.29.xlsx](#)
- [DB 1.30.xlsx](#)
- [DB 1.31.xlsx](#)
- [DB 1.32.xlsx](#)
- [DB 1.32a.xlsx](#)
- [DB 1.33.xlsx](#)
- [DB 1.34.xlsx](#)
- [DB 1.34a.xlsx](#)
- [DB 1.35.xlsx](#)
- [DB 1.35 original.xlsx](#)
- [DB 1.36.xlsx](#)
- [DB 1.36 original.xlsx](#)
- [DB 1.37.xlsx](#)
- [DB 1.37 original.xlsx](#)
- [DB 1.38.xlsx](#)
- [DB 1.38 original.xlsx](#)
- [DB 1.38a.xlsx](#)
- [DB 1.38b.xlsx](#)
- [DB 1.38c.xlsx](#)
- [DB 1.39.xlsx](#)
- [DB 1.40.xlsx](#)



MONKEYUSER.COM

# Our approach: structure, guidance and automation

1. Identify stakeholders / domain experts and agree about roles / expectations
2. Milestones with dates for each step – helps to keep the goal in mind
3. Help beyond just the scope of “coding”, but ...
4. ... know your audience!

5.

4	Version:											
5	Do not edit grey cells, they are generated automatically											
6												
7	Device Type	Device Type	Full Description of the signal	Short description of the signal	Not included in EPICS	If structured, see "Structures" sheet	Permission Level (Select)	Data type (Select)	Select integer from Int sheet	PLC Datablock	Byte offset	Bit off (default)
8	Full Name (Select)	(DO NOT EDIT!)		(max. 40 characters) Will enter the PV's DESC field.								
9												
10												
11	Device Type*	Device	Description*	Tag/name	Skip	Structure	Permission*	Type*	Int	DB*	Byte*	Bit
12	Example of AQ											

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	Type	No.	Relay Name	Assignment	EPICS Record Name	I/O comment	ON	OFF	ON Severity	OFF Severity	HIGH	Default	SCAN	Archive	Alarm
2	Shared Relay	1	E1	CPU1	KLY:SW:TEMP:ILK	Klystron thermo switch	OK	Interlock	NO_ALARM	MAJOR			1 second	Monitor	NO_ALARM
3	Shared Relay	2	E2	CPU1	KLY:IP1:PRS:ILK	Klystron ion pump 1 interlock	OK	Interlock	NO_ALARM	MAJOR			1 second	Monitor	NO_ALARM
4	Shared Relay	3	E3	CPU1	KLY:IP2:PRS:ILK	Klystron ion pump 2 interlock	OK	Interlock	NO_ALARM	MAJOR			1 second	Monitor	NO_ALARM

6. Automate “everything”!

# Tools are already out there

- No need to reinvent the wheel
  - We tried Visual Basic ...
  - ... and we stick to Python ☺

```
shared_memory_map_template.xlsm - Module1 (Code)
(General) MakeBase

Sub MakeBase ()
    ' Create text files
    Dim dbName As String
    Dim reqFile As String

    ' Obtain version number of this signal list
    Worksheets("History").Activate
    colVersion = Rows(1).Find("Version", LookIn:=xlValues, LookAt:=xlWhole).Column
    latestRow = Columns(colVersion).End(xlDown).Row
    latestVersion = Cells(latestRow, colVersion).Value

    Worksheets("Basic").Activate
    dbName = ActiveWorkbook.Path & "\" & Cells(5, 2) & ".db"
    reqFile = ActiveWorkbook.Path & "\" & Cells(5, 3) & ".req"
    archiveFile = ActiveWorkbook.Path & "\" & Cells(5, 4) & "_archive.xml"
    alarmFile = ActiveWorkbook.Path & "\" & Cells(5, 5) & "alarm.xml"

```

```
617
618     try:
619
620         # Error check: duplication of byte offsets
621         if line[plcdb_i] in DBs:
622             if ((line[byte_i] in DBs[line[plcdb_i]][1]) and not line[type_i] == 'BIT'):
623                 raise ValueError('Byte offset ' + line[byte_i] + ' in ' + line[plcdb_i] + ' already exists!')
624         else:
625             DBs[line[plcdb_i]] = ['', []]
626
627         # Assign record type
628         if (args.mode == 'r'):
629             if (line[type_i] == 'BIT'):

```

Generate EPICS database and autosave request file.

```
11 os.system('python parse_signals.py -i %s -m s' % sp_pt)
12 os.system('python parse_signals.py -i %s -m r' % rb_pt)
13
14 os.system('python generate_RB_to_SP_database.py')
15
```

## Project in numbers

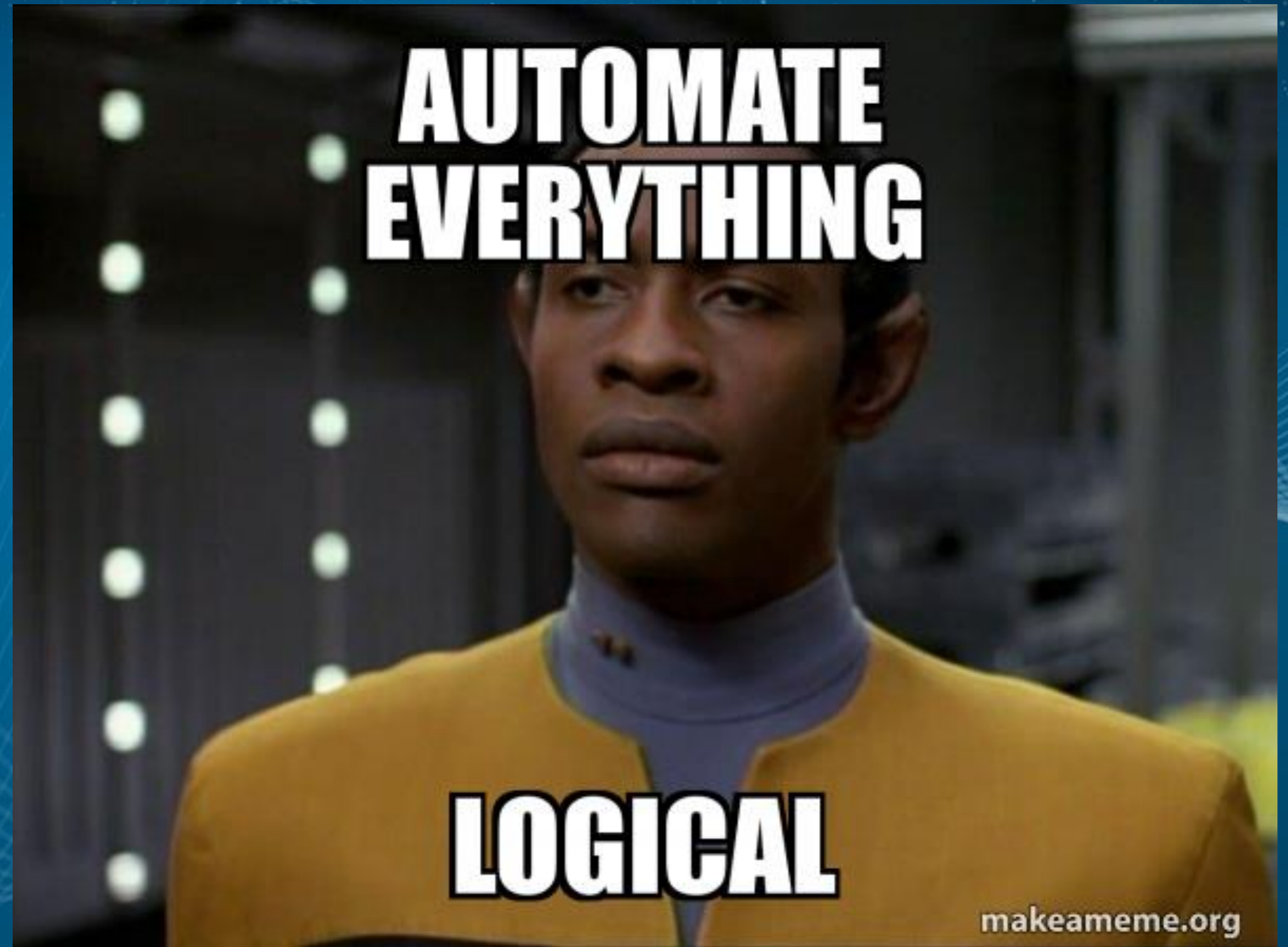
- Almost 50 versions after the “final” version of signals :D
- 9000 variables / records
- Scripts to generate and error check **database, archiver and alarm server configuration took 10% of effort**
- In **half year** we managed to define interfaces, implement all the business logic required on the EPICS level, prepare IOC app, create GUIs, configure central services ...

## Why are we doing it?

- Single interface / document between field experts and control system engineers
- Clear roles and expectations
- One input, multiple outputs in a single, automated step – **saves a lot of time**
- Added features: error checking, statistic ...
- “To err is human”, so if no human is involved = **“no” errors in the database**

# EPICS

## Rules, structure and possible



**EPICS**  


**Thank you**