Systematic production of beamline and other turn-key control systems

Gasper Pajor

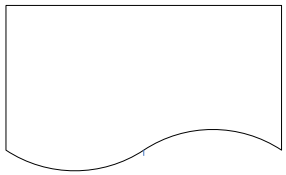
gasper.pajor@cosylab.com

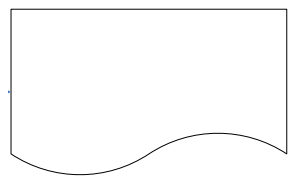
Turn-key Systems for Accelerators

- Each project has its own specific requirements
  - OS, platform, naming convention, etc.
- Turn-key systems often share some parts
  - Yet are specific -> solution recycling gets harder
  - Versioning can turn into a mess

- The Goal: the set of tools for
  - Easier but stricter version management
  - Efficient composition of a complex systems
    - Including new development
    - Controlled reuse (using well-tested solutions -> less testing time)
    - Centralized configuration
    - Repeatability
  - Eliminating the human factor wherever feasible

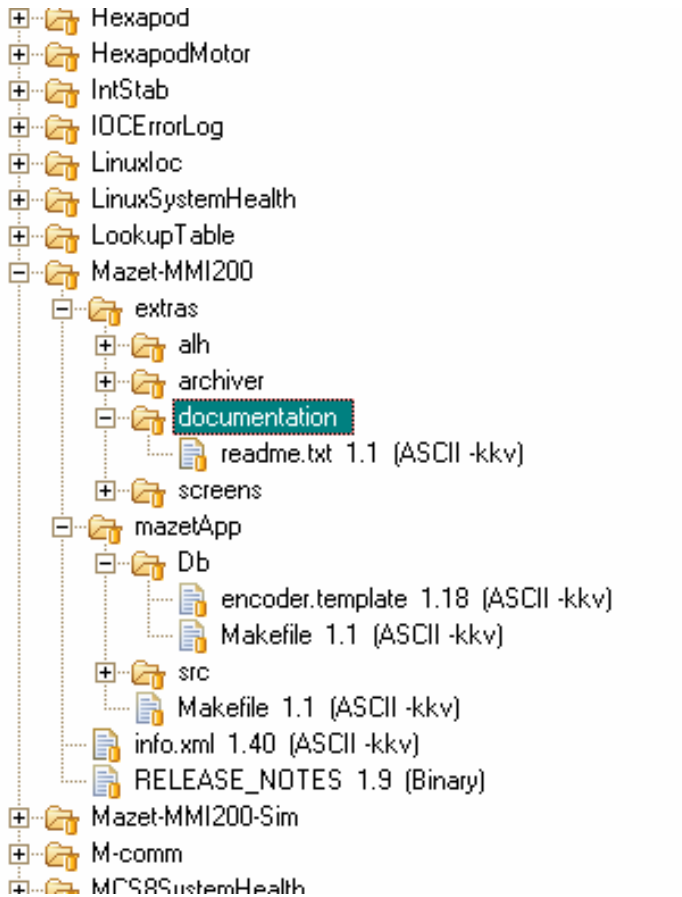
Design, a.k.a. The Big Picture





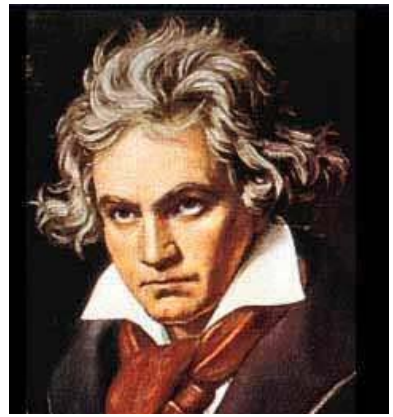
Generic Component

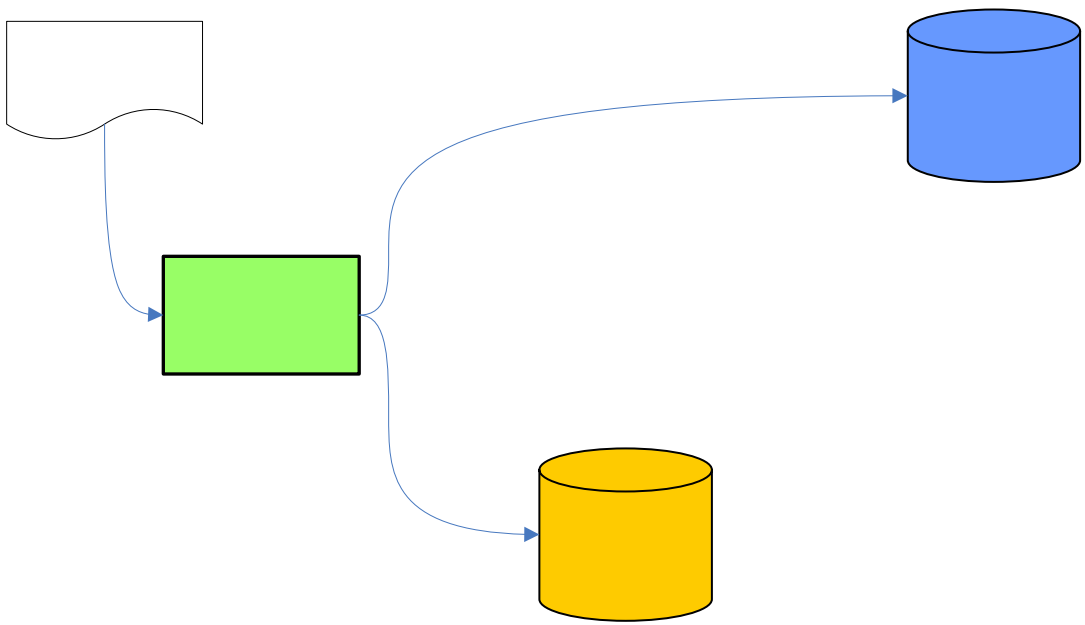
- Generic component consists of
  - Files (CVS location)
    - Well defined dir structure
  - Meta information
    - Configurable parameters
    - Written in info.xml
    - Transferred into Component SQL
- Typical components
  - EPICS base applications
  - Asyn driver
  - Other EPICS modules



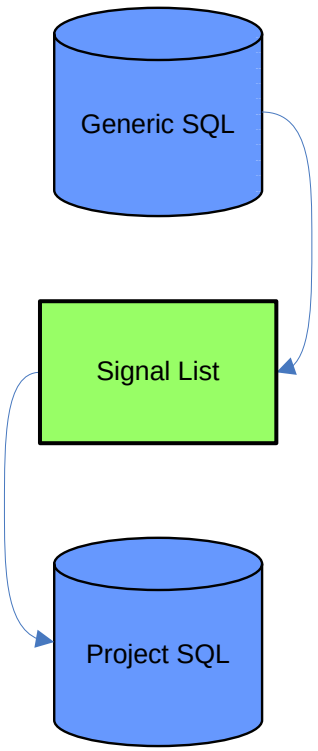
Orchestrator

- Version control enforcing tool
- Checks info.xml for consistency
- Ensures the CVS synchronization
  - Commits to CVS if necessary
- Tags the CVS
- Inserts the metadata into “Generic SQL”

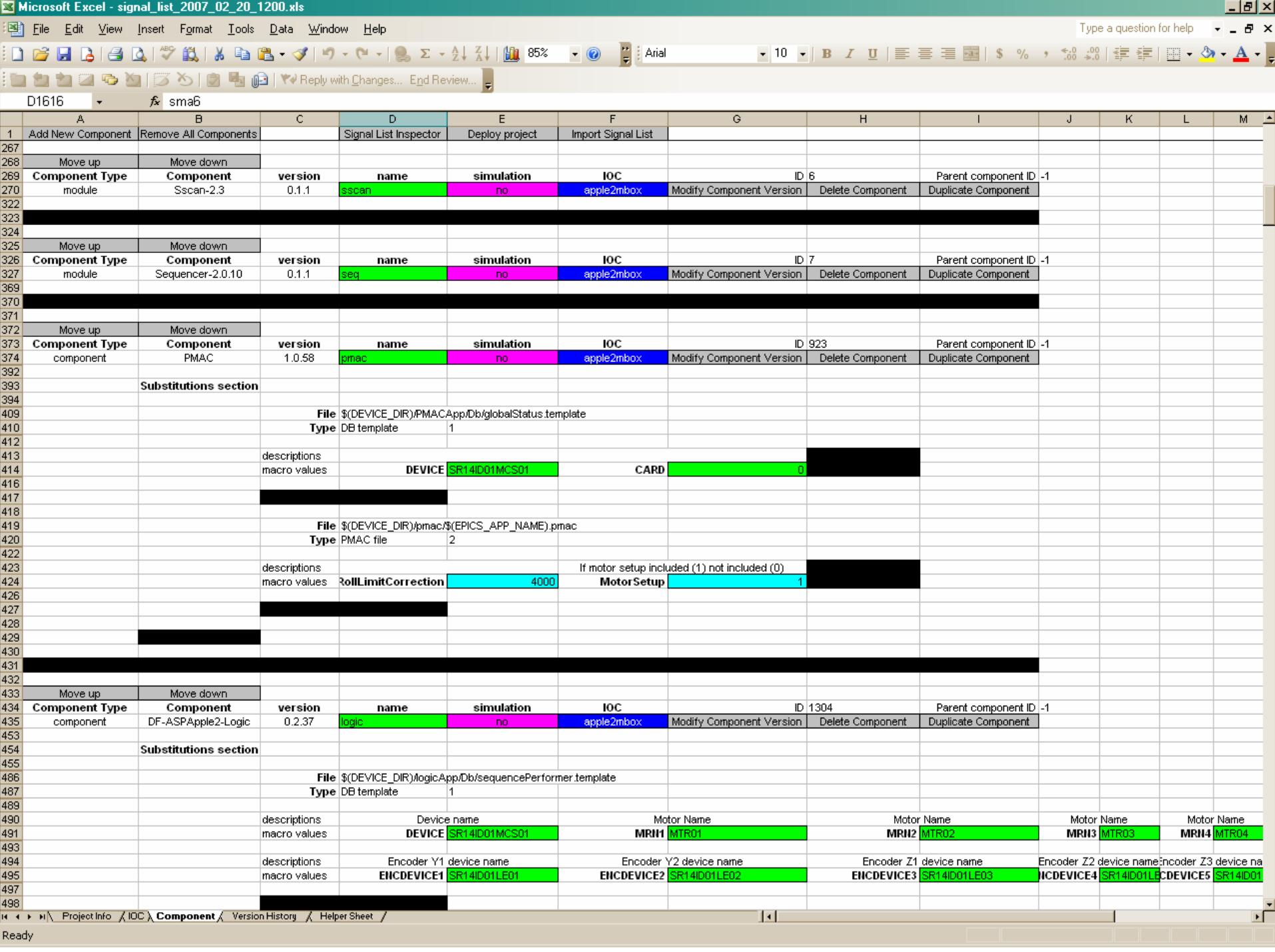




Signal List

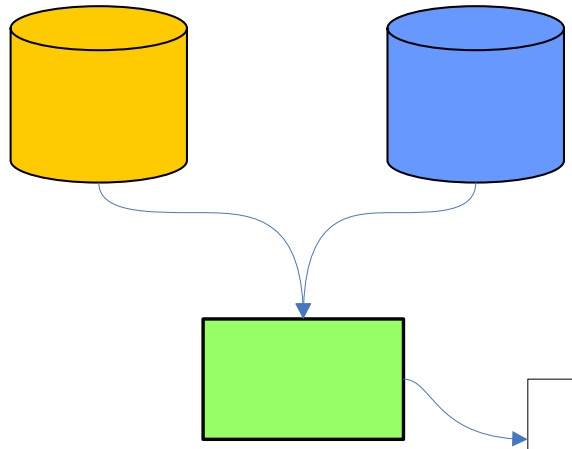


- Not really a classic signal list
- Configuration tool
  - Done in excel – a lot of convenient spreadsheet functionality out of the box
- Allows user to
  - instantiate the components from “Generic SQL”
  - configure the components
  - configure the IOC parameters
    - network settings, required libs, …
  - puts the accumulated data to “Project SQL”
    - Project ID
- Consistency checking


Microsoft Excel - signal_list_2007_02_20_1200.xls
File Edit View Insert Format Tools Data Window Help
Type a question for help
Arial 10
Reply with Changes... End Review...
D1616 sma6
Add New Component
Remove All Components
Signal List Inspector
Deploy project
Import Signal List
Move up
Move down
Component Type
Component
version
name
simulation
IOC
ID 6
Parent component ID -1
module
Sscan-2.3
0.1.1
sscan
no
apple2mbox
Modify Component Version
Delete Component
Duplicate Component
ID 7
module
Sequencer-2.0.10
seq
ID 923
component
PMAC
1.0.58
pmac
Substitutions section
File $(DEVICE_DIR)/PMACApp/Db/globalStatus.template
Type DB template 1
descriptions
macro values
DEVICE SR14ID01MCS01
CARD 0
File $(DEVICE_DIR)/pmac/$(EPICS_APP_NAME).pmac
Type PMAC file 2
If motor setup included (1) not included (0)
RollLimitCorrection 4000
MotorSetup 1
ID 1304
component
DF-ASPApple2-Logic
0.2.37
logic
File $(DEVICE_DIR)/logicApp/Db/sequencePerformer.template
Type DB template 1
Device name
Motor Name
MRN1 MTR01
MRN2 MTR02
MRN3 MTR03
MRN4 MTR04
Encoder Y1 device name
Encoder Y2 device name
Encoder Z1 device name
Encoder Z2 device name
Encoder Z3 device na
ENCDEVICE1 SR14ID01LE01
ENCDEVICE2 SR14ID01LE02
ENCDEVICE3 SR14ID01LE03
ENCDEVICE4 SR14ID01LE
ENCDEVICE5 SR14ID01
Project Info IOC Component Version History Helper Sheet
Ready

Weldor

- Tool for automatically welding everything together
- 6 steps:
  - Fetching data from Project SQL
    - Based on project ID
    - Filling the data model
  - Fetching files from CVS
    - Particular CVS tag of each component
    - Putting the files in appropriate dirs
  - Applying naming convention
    - Replacing generic names
  - Welding
    - Modifications of the files based on data from Project SQL
  - Invoking the next stage
    - Clean-up
    - Preparing configuration files for deploying
- All parts are pluggable
  - To accommodate alternate implementations
  - Subversion instead of CVS in next couple of month



Naming Convention

- Problem: How to cover most of naming conventions used in labs with
  - pattern for all generic components' record names
  - name-changing algorithm
- Solution:
  - Pattern: $(DEVICE)$(PROPERTY)$(TYPE)
  - In generic component: $(DEVICE)$(VOLTAGE)$(GET)
  - In instance for Lab A: $(DEVICE):VOLT_MONITOR
  - In instance for Lab B: $(DEVICE)::Get-Voltage
    - $(DEVICE)
      - usually left in this form to be replaced later by substitutions file
    - $(PROPERTY)
      - a (unique) macro for propery name, which is replaced during naming convention
    - $(TYPE)
      - one of $(GET), $(SET), $(STATUS), $(CMD) or $(NONE), which are changed into their respective strings for particular LAB
  - Extensive use of regular expressions

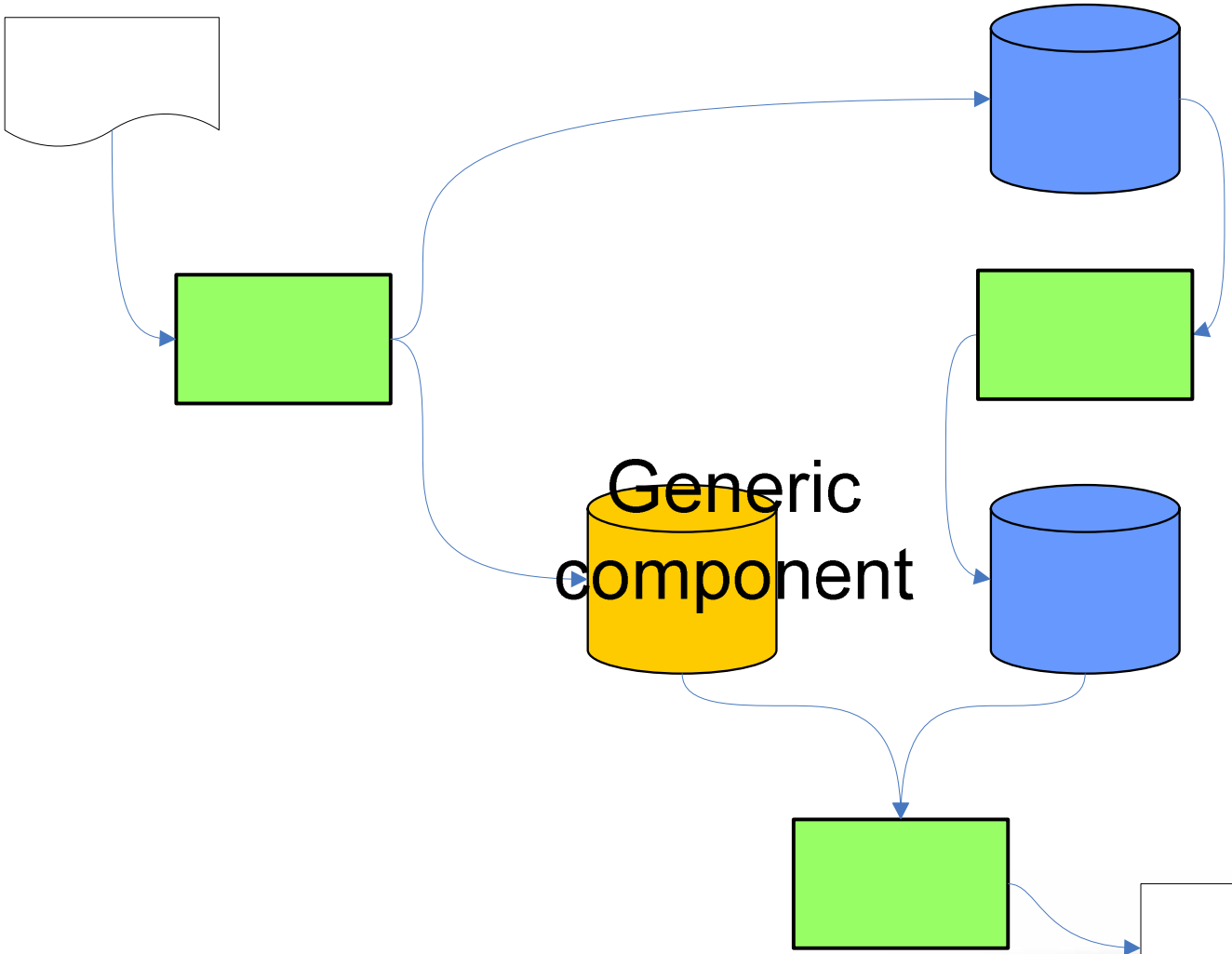
By-products

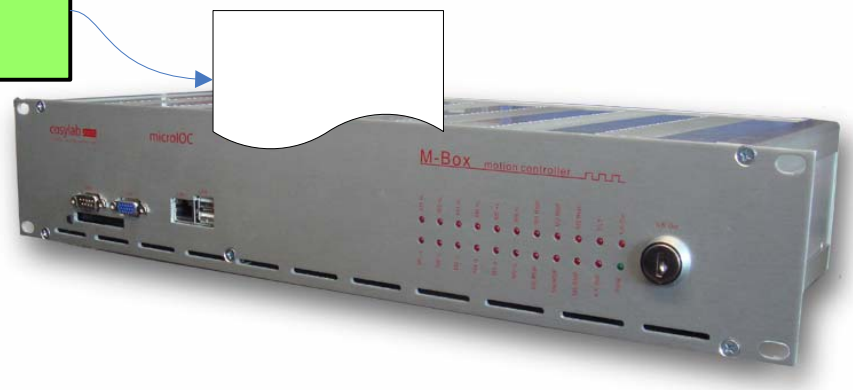
- Miner - Data mining tool
  - Versions of certain project
  - Configuration/component diff
- PMAC configuration Generator
  - Alongside EPICS configuration (same process)
  - Making PMAC configuration manageable
    - No variable overlapping
    - Use of templates
- Il Documentatore
  - Naming convention on the documents

Summary

- A set of processes and tools for
  - Development
  - Centralized configuration
  - Deployment
  - Project Management
- The use of these tools leads to
  - Higher quality
  - Better efficiency
  - Less possibilities for human errors
  - Controlled re-use of well tested components
    - More time for system-wide quality assurance

Thank you!


Generic
component


cosylab
microIOC
M-Box
motion controller