User Autonomy

Internal software collaboration at
MAX IV

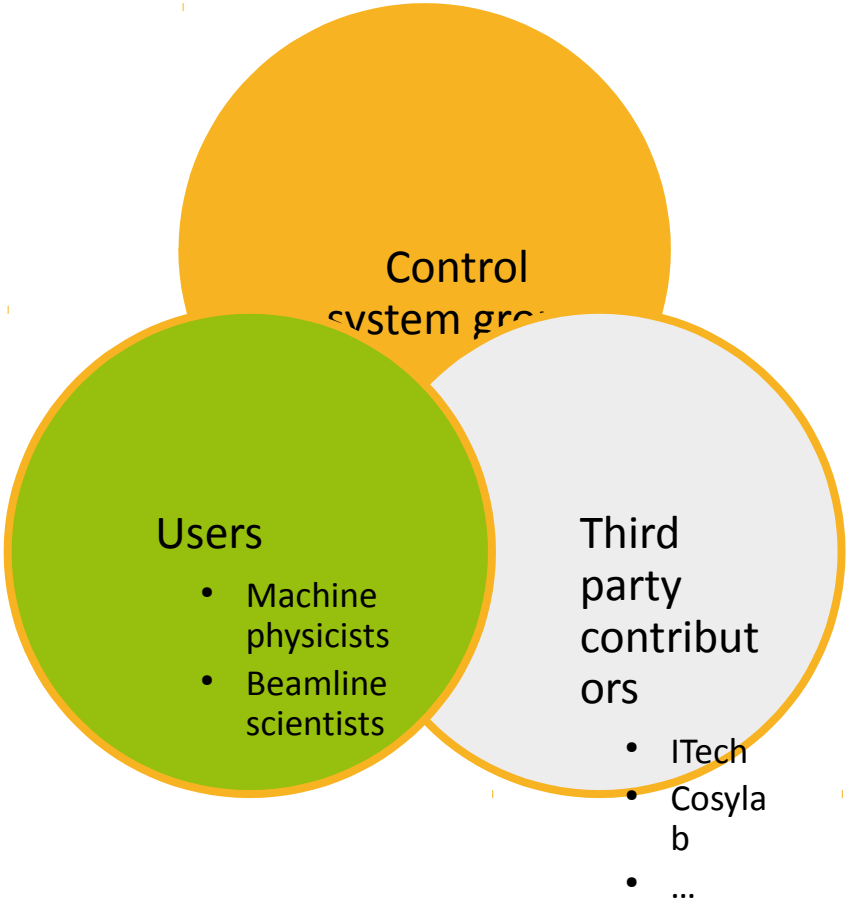
User Autonomy - overview

- Motivation and vision
- How we do it, philosophy and tools
- Some examples
- Future plans

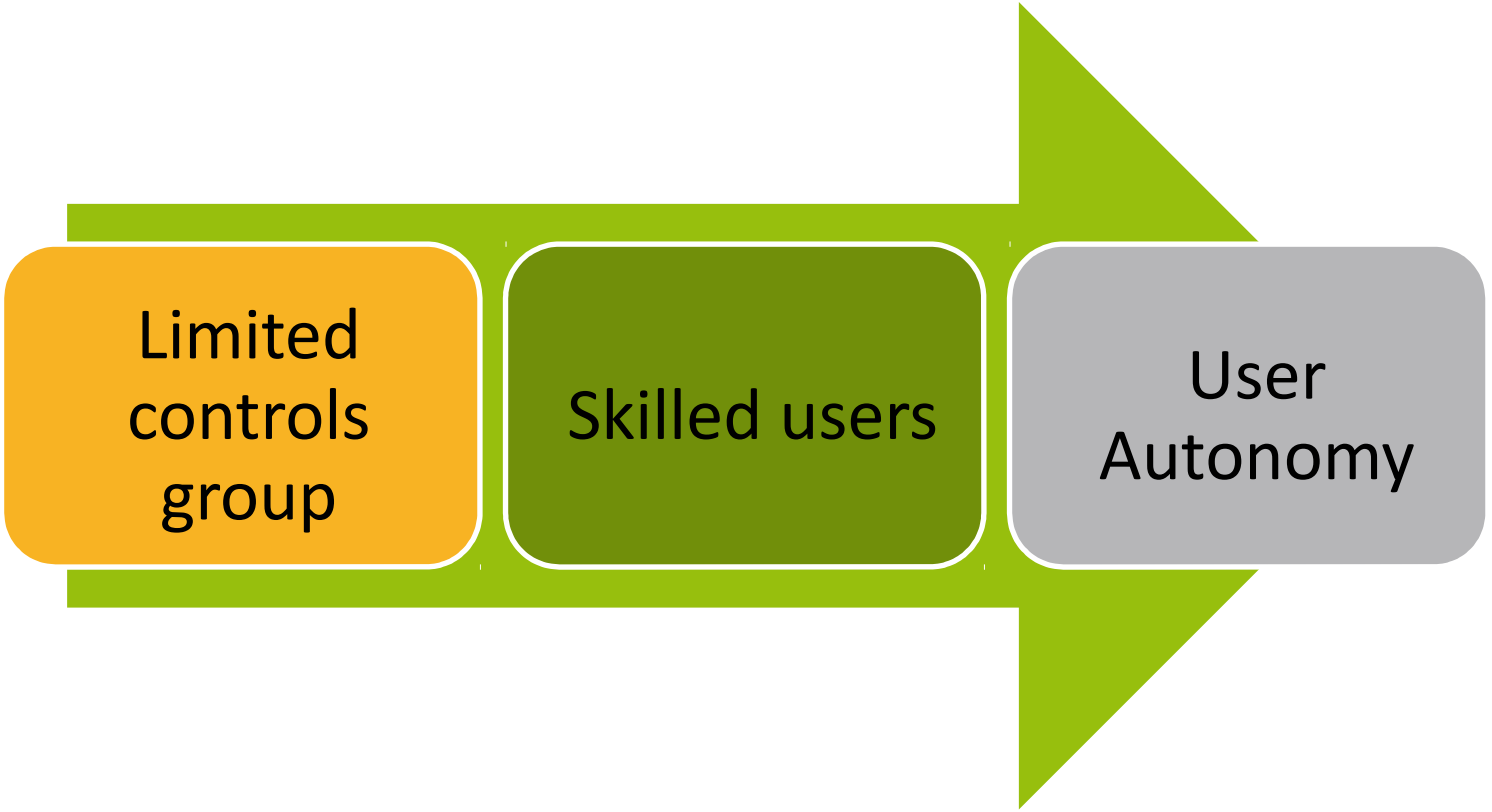
Mirjam Lindberg and Andreas Persson

On behalf of MAX IV KITS group

Achieving an open control system



User Autonomy – why we want it



User Autonomy – reaching out

User Autonomy – working together

User Autonomy – our vision

What do we hope to gain from working together?

- Utilize the Maxlab spirit
- Freedom but not anarchy
- Contribution of knowledge and skills

User Autonomy - pitfalls

Risks

- Less control in the control system
- Rogue developers
- Non-standard solutions

Mitigate by planning ahead and having a clear process

User Autonomy – how we do it

Come to the dark side,
we have cookies!

(and coffee)

User Autonomy – how we do it

KITS Café

- Install party
- Tango
- Coding Dojo
- PyTango
- Taurus
- Matlab-Tango binding

User Autonomy – how we do it

Support from the KITS group

- Explaining concepts
  - Tango
    - PyTango
    - Matlab-binding
  - Sardana
  - Taurus
- Code review
  - Ideally before deployment
- Integration and packaging

```
#!/usr/bin/python
from PyTango import *
import time

samplex = DeviceProxy('sys/test/samplex')
detx = DeviceProxy('sys/test/detx')
sampley = DeviceProxy('sys/test/sampley')
dety = DeviceProxy('sys/test/dety')

sampleposition = 0
maxvalue = [0,0]
maxpos = [0,0]
number = [0,0]

detx.Stop()
dety.Stop()
samplex.Velocity=100
samplex.Acceleration=1000
sampley.Velocity=100
sampley.Acceleration=1000

def move(actuator, position):
        actuator.Position = position
        while (actuator.State() != DevState.ON ) :
                time.sleep

def read(detector):
        detector.Start()
        time.sleep(1)
        detector.Stop()
        return detector.value
```

User Autonomy – how we do it

Easily accessible environment

- NX Server with standardized MAX IV environment
- Personal accounts for **all** employees
- RPM packages for in-house and third party software
- Tango Host for experimentation
- Simulated devices
  - SimuCounter, SimuMotor, etc.
- Source code repositories for employees

User Autonomy – how we do it

Groupware

- Wikis
  - HOWTOs
  - Project specifics
- Gitorious
  - Git repositories
  - Groups and projects
  - Code review
  - Pull requests

RPM packaging - details

Overhaul of package building process

- Build automation with Jenkins
- .spec file (or setup.py) in source code repository
- Fedora build tools (pyrpkg, mock, …)
- EL5 (selected packages), EL6, Fedora 18

Next

- Release strategy
- Update public yum repositories

User Autonomy - examples

- User projects
  - Gun test slit control
    - Trinamic Tango DS (python)
  - MAX IV machine physicists
    - Matlab/Tango scripts with the virtual accelerator
  - Maxlab beamlines
    - Machine status for mobile devices (not _yet_ with Tango)
    - Version control
- GUI Prototyping
  - user presents requirements through a Taurus prototype

User Autonomy – future plans

- A defined process for integrating user contributions
  - Code review
  - Version control
  - Testing
  - Build automation
  - Packaging
- Enable users to build custom GUIs with our widgets
  - Following the example of ALBA
- KITS Café Hackathons

User Automation - conclusion

Open control system with a safe landing

Thanks for listening!