

# Talking ITIL for Agile Folks (learning Standard Change)

R1

[robin.slomkowski@nokia.com](mailto:robin.slomkowski@nokia.com)

# Talk the Right Language!

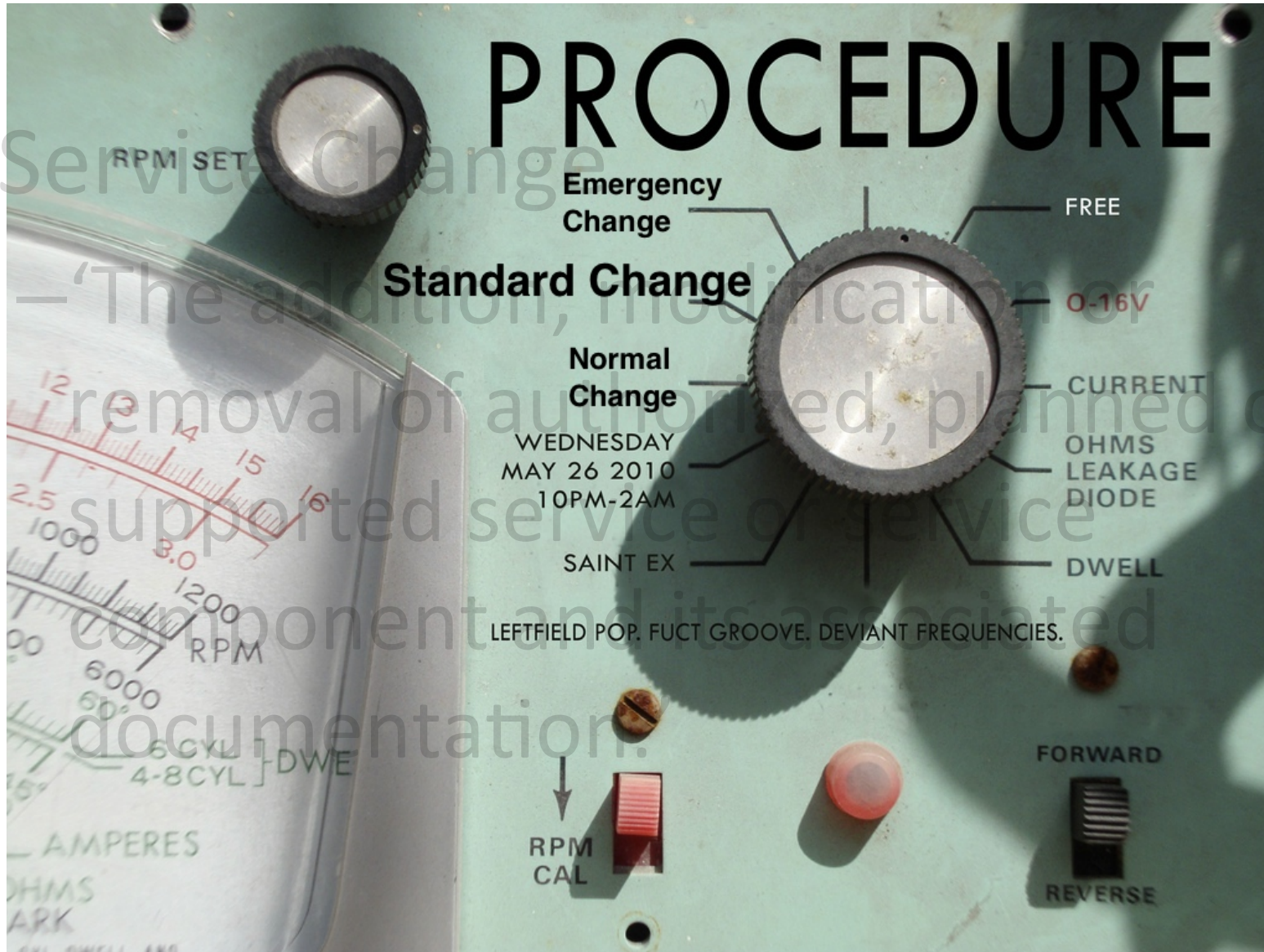
- Stop Fighting, they are just requirements!
- They have a big book that defines their language, but that is because it is complicated.



# Talk the Right Language!



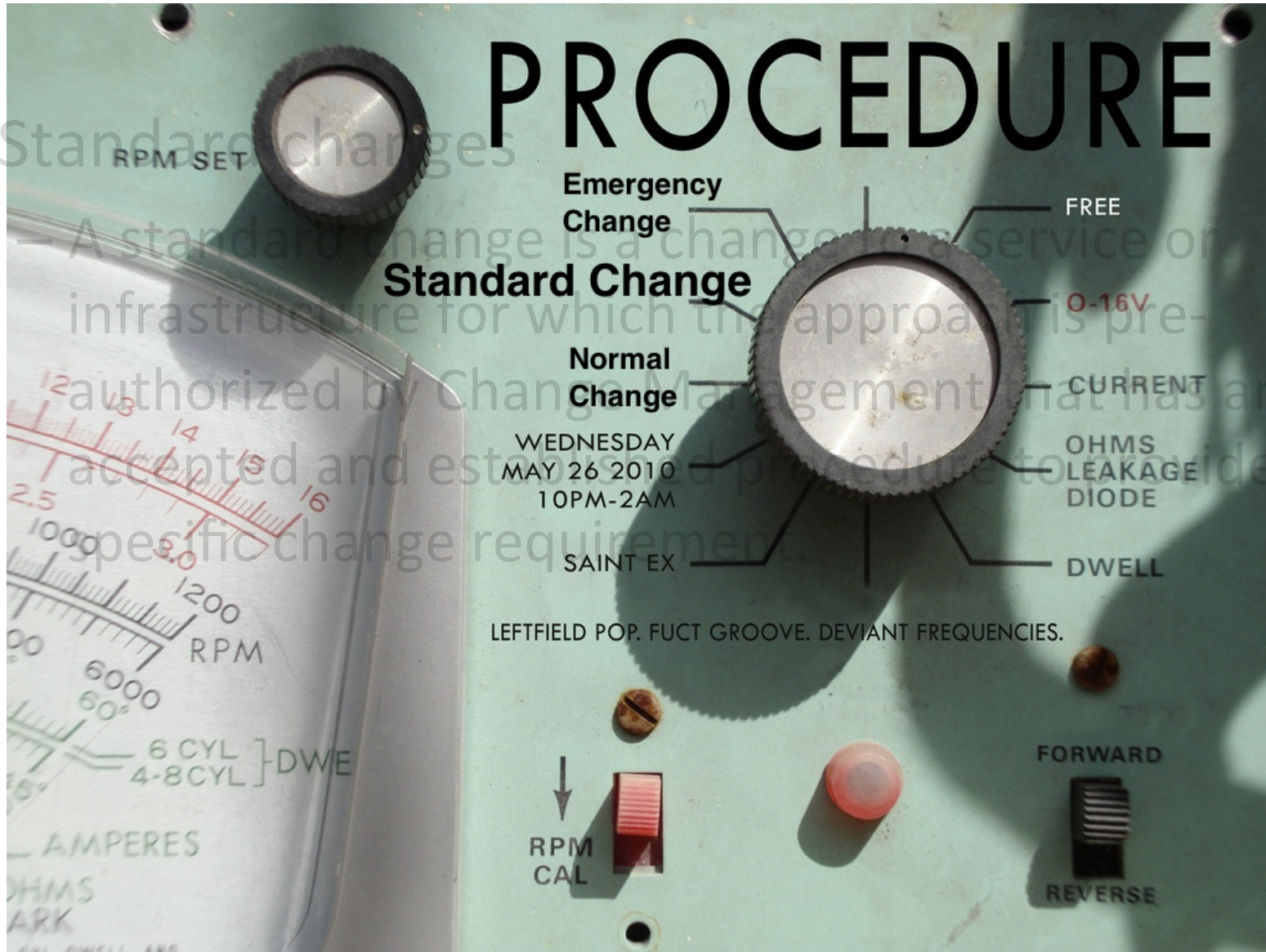
# Standard Change



- Service Change – 'The addition, modification, removal of authorized, planned or supported service or service component and its associated documentation.'

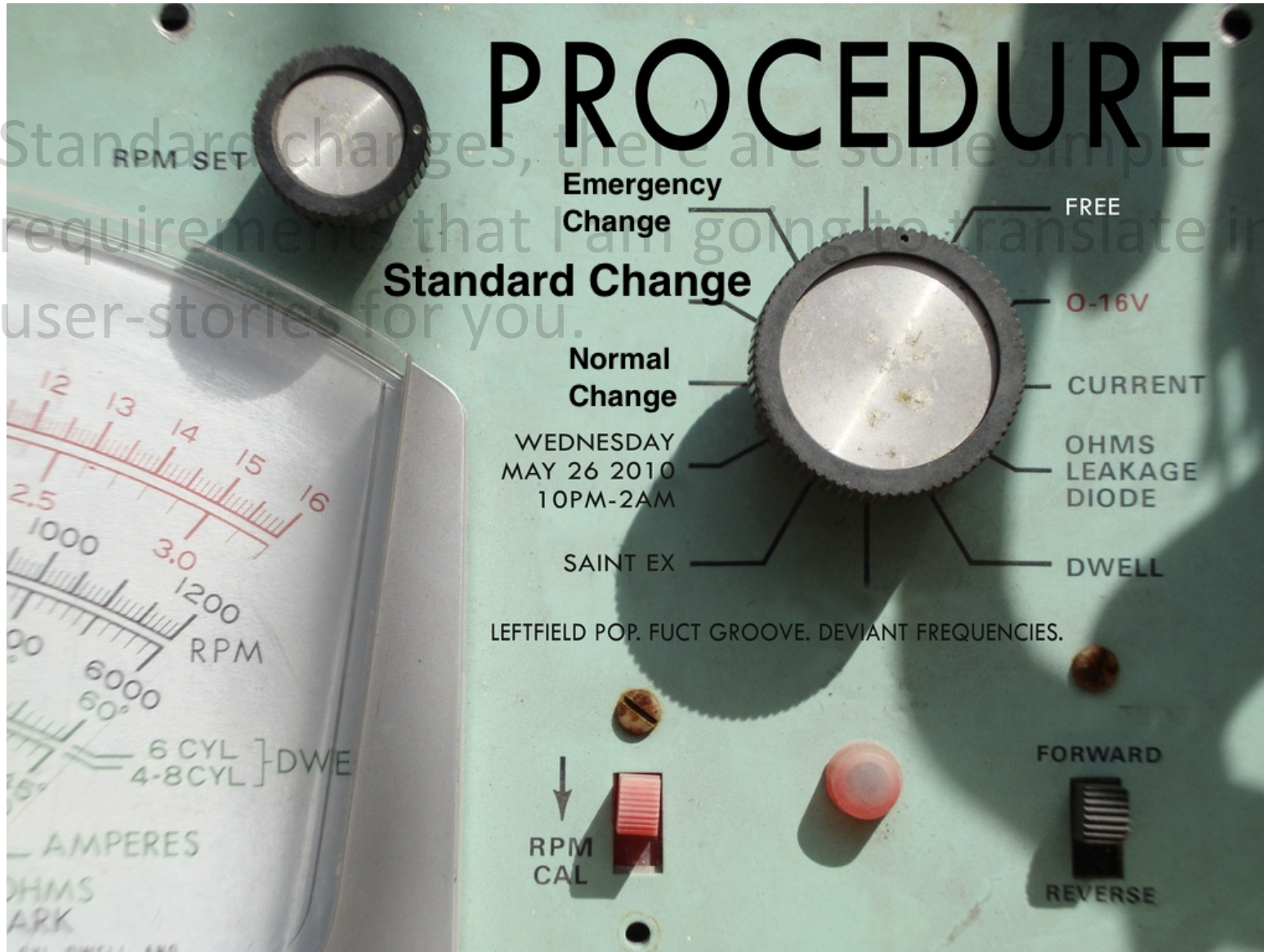
# Standard Change

- Standard changes  
– A standard change is a change to a service or infrastructure for which the approach is pre-authorized by Change Management that has an accepted and established procedure to provide a specific change requirement.



# Standard Change

- Standard changes, there are some simple requirements that I am going to translate into user-stories for you.



There is a defined trigger to initiate the RFC



There is a defined trigger to initiate the RFC





# The tasks are well known, documented and proven



# The tasks are well known, documented and proven

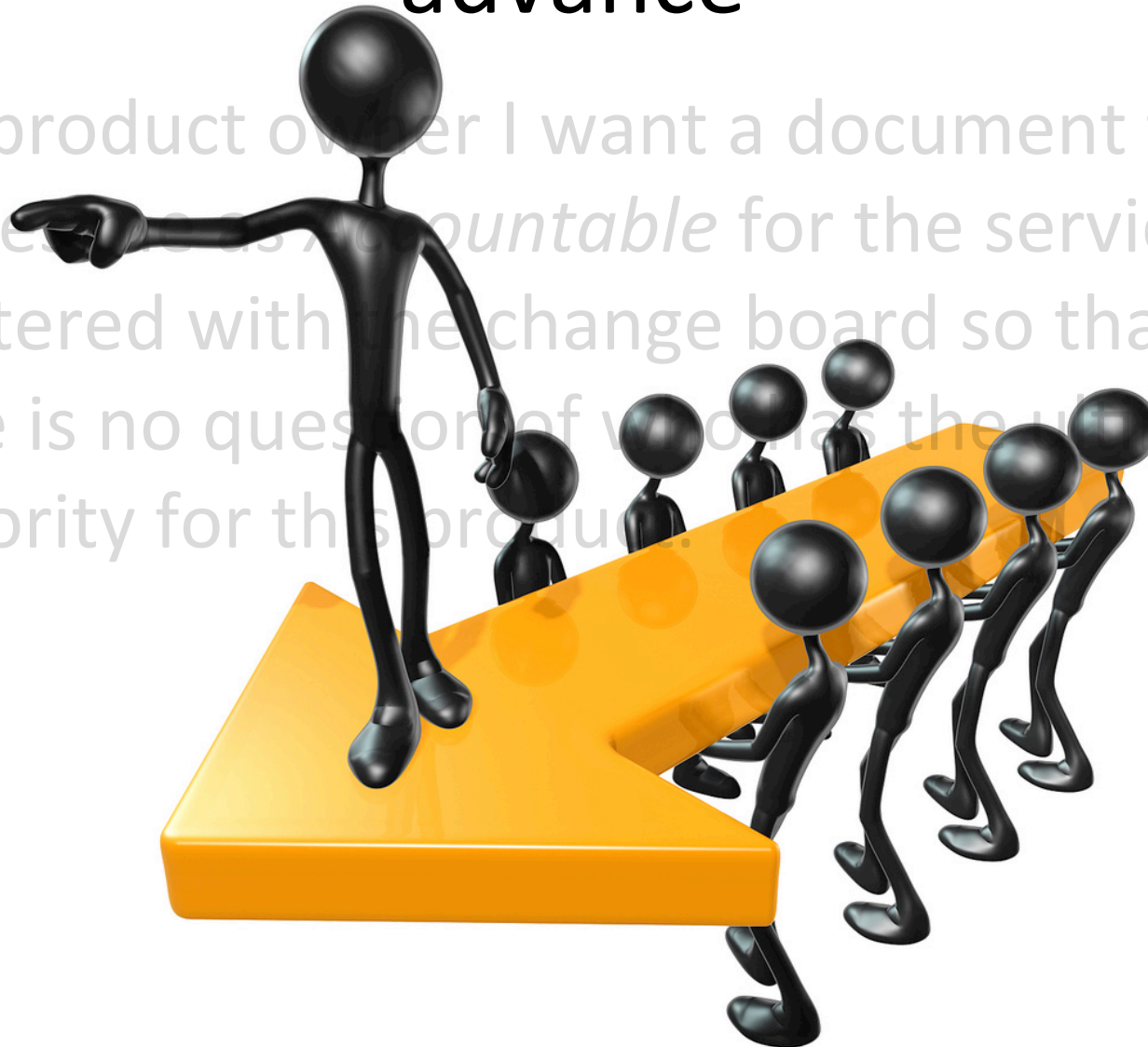


# The tasks are well known, documented and proven



# Authority is effectively given in advance

- As a product owner I want a document that defines *responsibilities* *accountable* for the service registered with the exchange board so that there is no question of *who* has the ultimate authority for this product.



# Authority is effectively given in advance

- As a product owner I want a document that shows that my release processes meet my company's process, legal, and/or other restrictions so that there can be no question or delays about going to production and getting the code out.



# Budgetary approval will typically be preordained or within the control of the change requester

- As a product owner I want tools to allow me to monitor the resource use or cost of my releases so that I can keep my company's budget predictable.



# The risk is usually low, and always well understood

- As  
sm  
ris  
fix



at  
the  
y

# The risk is usually low, and always well understood

- As  
cho  
be  
mi



at  
ong



# The risk is usually low, and always well understood

- As  
pro  
ou  
tha  
be



mated  
ust like  
em so  
ood

# What about WHEN it goes wrong?

- Your penalty for poor testing is bureaucracy, you lose your standards, change when it causes your users a poor experience



# What about WHEN it goes wrong?

- Your answer is that you will create a new test and integrate it all the way through your integration pipeline so you never make the same mistake twice and eventually learn to love your releases!



# Compliance & Agile as Friends!

