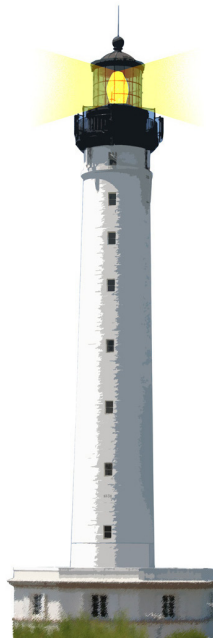


# About Validation

A pretext to talk about delegation of actions

S. Ducasse



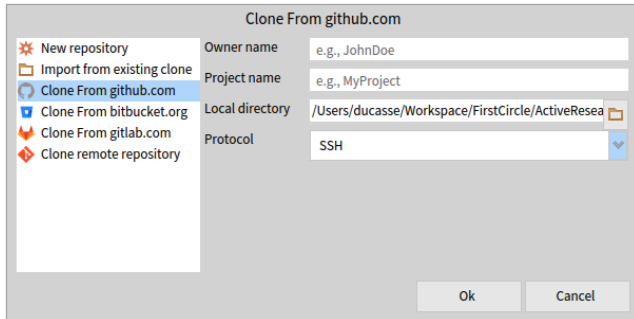
# Goal

- How can we navigate an instance tree?
- With optional states that can be skipped?



# The case: Validation

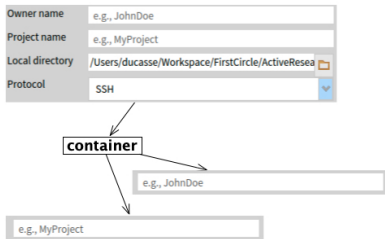
- We want to validate UI forms
- Nested components may want to validate or not their contents



The image shows a dialog box titled "Clone From github.com". On the left is a list of options with icons: "New repository" (star), "Import from existing clone" (folder), "Clone From github.com" (GitHub logo, highlighted), "Clone From bitbucket.org" (Bitbucket logo), "Clone From gitlab.com" (GitLab logo), and "Clone remote repository" (diamond). On the right are input fields: "Owner name" with the text "e.g., JohnDoe", "Project name" with the text "e.g., MyProject", "Local directory" with the text "/Users/ducasse/Workspace/FirstCircle/ActiveResea" and a folder icon, and "Protocol" with a dropdown menu showing "SSH". At the bottom right are "Ok" and "Cancel" buttons.



# A tree of instances



# A first design

- Any presenter can validate its contents.
- Per default does not nothing.

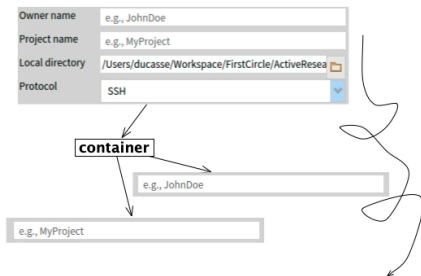
```
SpPresenter >> validate  
  ^ self
```

```
SpOptionPresenter >> validate
```

```
| report |  
report := SpValidationReport new.  
self layout children do: [ :presenter | presenter validate ]
```



# Flow's first design

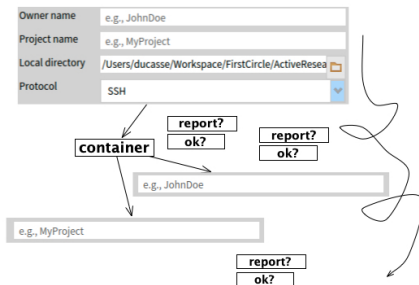


# Analysis

- We need to have a report to know if the validation failed or not.
- Should `validate` return a report?
- If `validate` returns a report then we have to return an ok report for anybody.
- We do not want to force a report for all the tree instances.



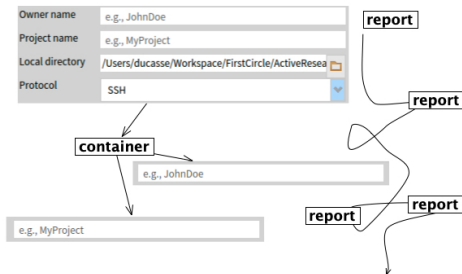
# Flow's first design





# Second design: provide an accumulator

Pass around a basket and let any sub instance decides if it wants to participate



## Second design: code

```
SpPresenter >> validateInto: aReport  
  ^ self
```

```
SpOptionPresenter >> validate  
  
  | report |  
  report := SpValidationReport new.  
  self layout children do: [ :presenter | presenter validateInto: report ].  
  ^ report
```



# Local and global together

```
SpTextInputFieldWithValidation >> validateInto: aValidationReport  
  self validate.  
  aValidationReport addAll: validationErrors
```

Each validating subcomponent

- gets the responsibility to fill up the report
- can bring its information to the report



# Conclusion

- Let the object decides if it wants to join a process but passing a container



A course by

S. Ducasse, L. Fabresse, G. Polito, and Pablo Tesone



Except where otherwise noted, this work is licensed under CC BY-NC-ND 3.0 France

<https://creativecommons.org/licenses/by-nc-nd/3.0/fr/>