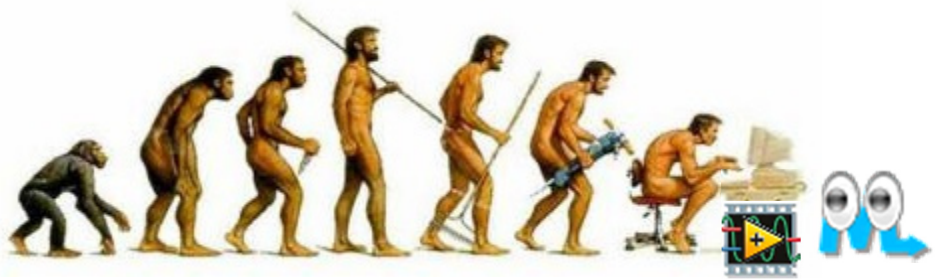
Darwin *appliqué à* ***LabVIEW*** *:*
l'évolution de la gestion des données



Evoluer pour s'adapter

La loi du plus fort appliquée au concept « mémorisation » du flux de données :

contrôle vers indicateur - Locale - Globale - FGV - AE – SEQ - DVR - OOP – SM - QDMH

Avant de commencer

- Faire évoluer la présentation en fonction de vous
- Vous pouvez m'interrompre
- Détailler en fonction de vos demandes

- Mes excuses pour les mots anglais = technique LV

La société MESULOG en bref

- Activité : Développement logiciel test et mesure
- Compétences : LabVIEW (Windows, RT, PDA, DSC, FPGA),
  TestStand
  VeriStand
- Localisation : Grenoble (Moirans, 38)
- Partenaire National Instruments (2001)
- Développeurs certifiés LabVIEW et TestStand

NATIONAL INSTRUMENTS Silver Alliance Partner

NATIONAL INSTRUMENTS LabVIEW CERTIFIED ARCHITECT

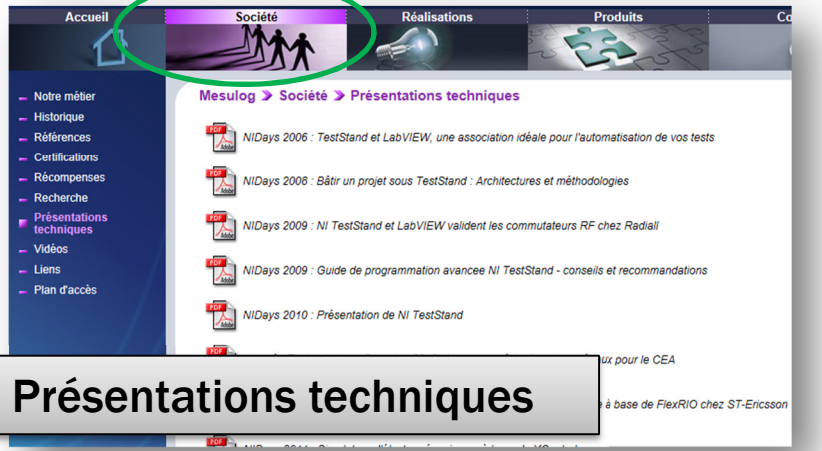
NATIONAL INSTRUMENTS CERTIFIED NI TestStand ARCHITECT

La société MESULOG en bref

- **www.mesulog.fr**
  - Réalisations (article utilisateur)

Réalisations

- Présentations techniques
  - LabVIEW
  - TestStand
  - VeriStand

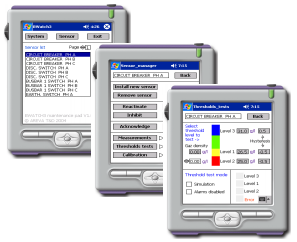


Société -> Présentations techniques

La société MESULOG en bref

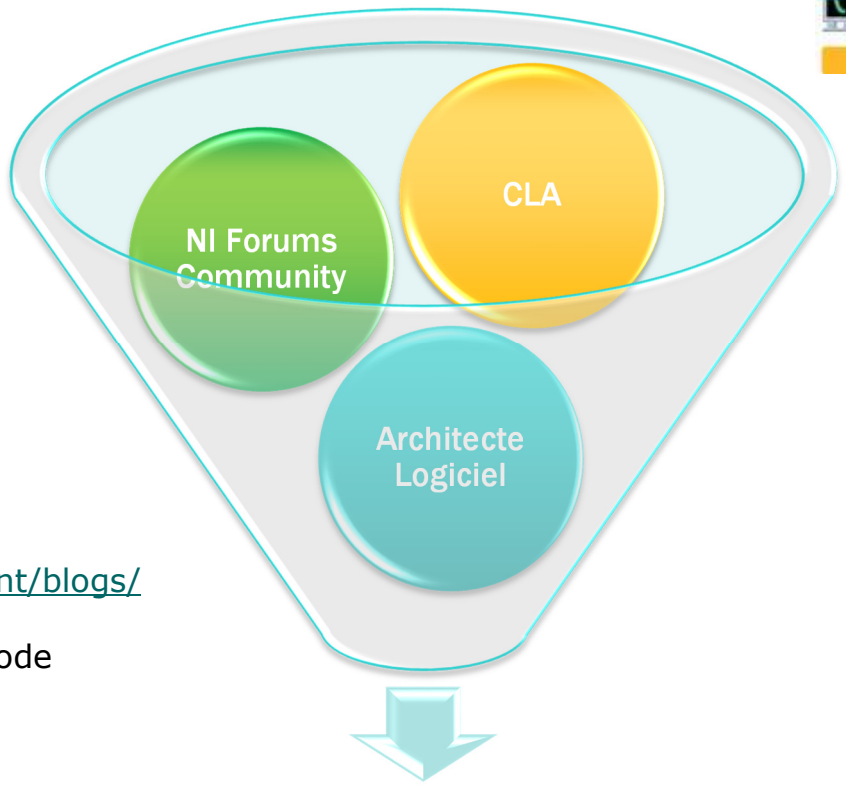
- **Ils nous ont fait confiance :**

  - ALSTOM Grid
  - AREVA NP
  - CEA (Commissariat à l'Énergie Atomique)
  - CETIAT (Centre Technique des Industries Aérauliques et Thermiques)
  - CNES (Centre National d'Études Spatiales)
  - CNRS (Centre National de la Recherche Scientifique)
  - EDF
  - HONEYWELL Security
  - LNE (Laboratoire National d'Essais)
  - ONERA
  - PECHINEY
  - RADIALL
  - RENAULT
  - STMicroelectronics
  - THALES Alenia Space
  - THALES LCD
  - ...



Le présentateur

NATIONAL INSTRUMENTS LabVIEW CERTIFIED ARCHITECT



Community blogue
https://decibel.ni.com/content/blogs/Luc_Desruelle

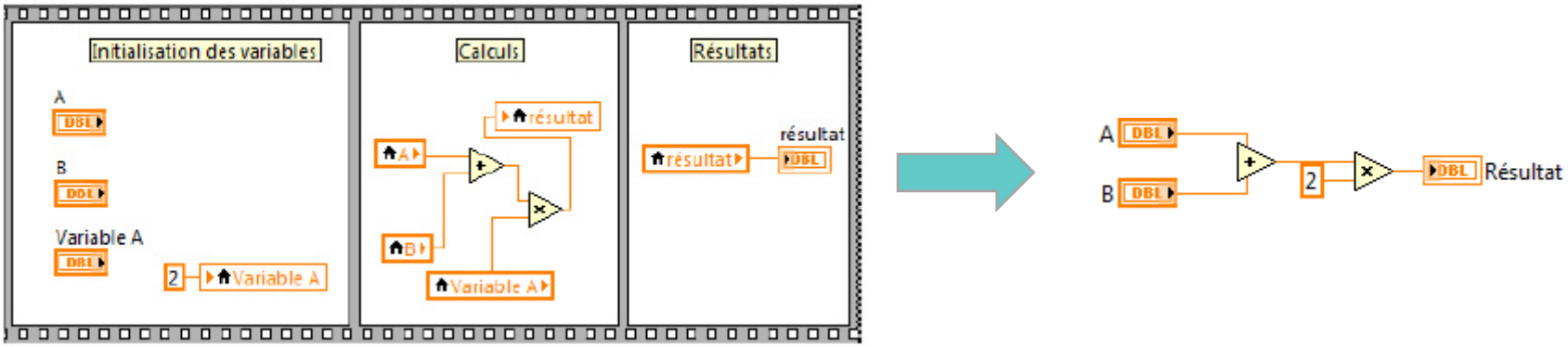
- Gif animé pour éviter du code
- Fenêtre pas rectangulaire
- Gestion IHM
- Projet LabVIEW
- OOP
- Modbus
- ….

Luc DESRUELLE

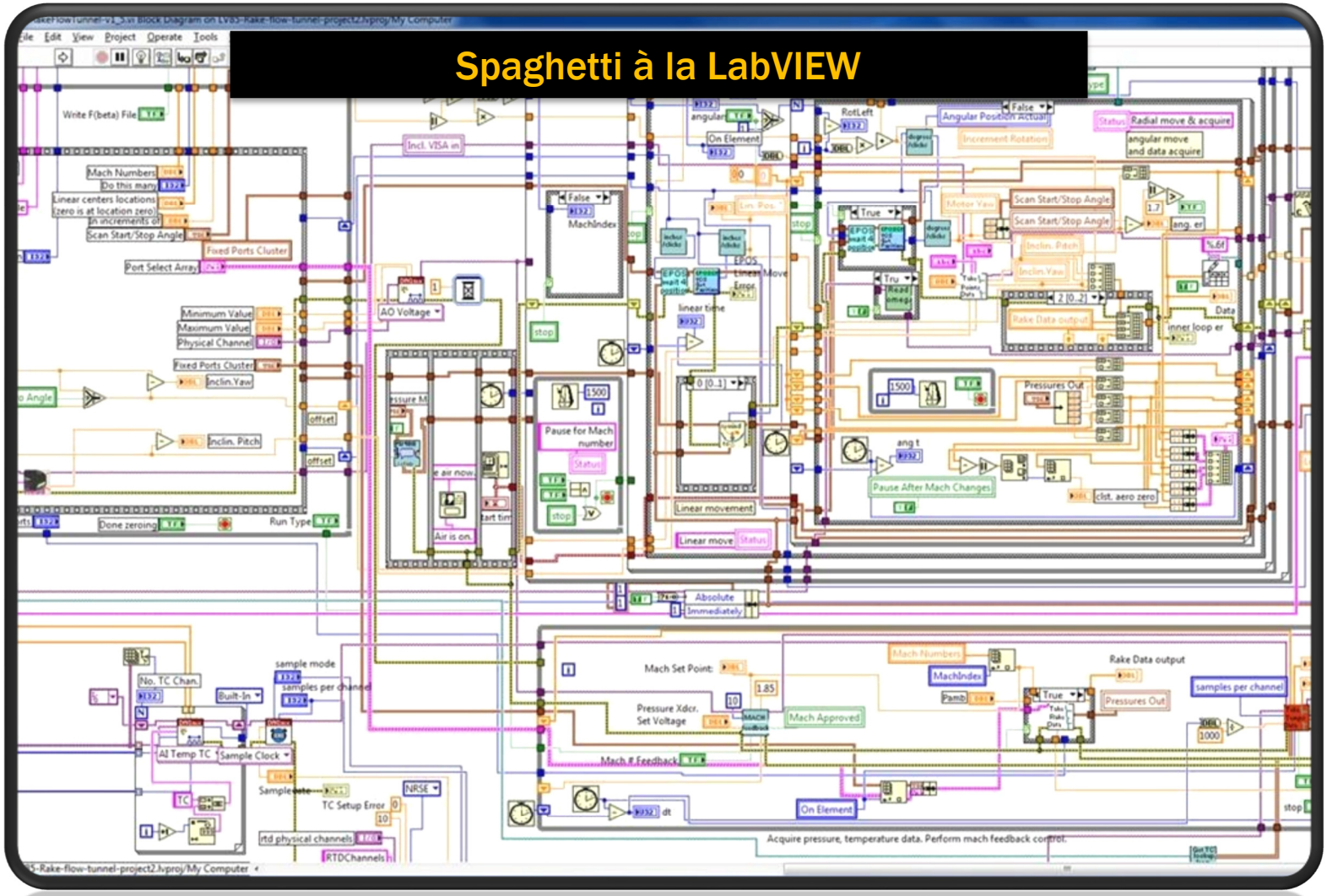
Le présentateur

- **Co-auteur Livre « LabVIEW : programmation et applications »**
  - 3iéme Édition, Dunod 05/2015
  - Lien internet du livre
  - Auteurs : F. COTTET + M. PINARD + L. DESRUELLE
    - Chapitres 1 et 2 : prise en main de l'outil avec des exemples simples
    - Chapitre 3 : programmation avancée avec des techniques et architectures permettant au code d'être maintenable, évolutif, documenté et performant.
    - Chapitres 4, 5 et 6 : acquisition, analyse et présentation des données.

- Certaines « erreurs » sont acceptables… si « rares »



Le code LabVIEW le plus célèbre



- Quelles sont les erreurs (programmation LabVIEW) ?

Erreur des Spaghettis sous « LabVIEW »

1. **Style, écriture du code**
   - Pas de style!
   - Documentation

2. **Technique, Gestion des données**
   - Trop de locale – Globale
   - Flux de données

3. **Architecture, Structure application (Framework) :**
   - Absence modèle de conception
   - Pas gestion d'erreur, Gestion arrêt du logiciel?

4. **Gestion Projet**
   - Version, SCC, bug, tâche
   - Test, analyse

Nous n'aborderons pas

- **Nous n'aborderons pas la gestion « projet »:**

  Présentation LUGE 2013 et NIDays 2014 :

  Les outils qui nous veulent du bien

  Plus de temps pour développer en LabVIEW

- **Nous n'aborderons pas la gestion « style »**

  Présentation LUGE 2016 et NIDays 2017:

  Avoir du style avec LabVIEW

Nous aborderons

- **Les différentes techniques de gestion des données**
  - Mémoriser une donnée
  - Réutiliser une donnée déjà produite
  - Quand, comment, pourquoi utiliser certaines techniques ?

Chapitre 3 du livre LabVIEW…. 

- **Pas les techniques d'échange de données :**
  - Entre processus;
  - Entre boucles;
  - Entre applications.

FIFO, notifier, Variable partagée, TCP, UDP, ActiveX, .Net, Network streams, DMA, Web service…

Chapitre 6 du livre LabVIEW…. 

Sommaire : Darwin appliqué à LabVIEW

I] Où trouver de l'aide ?

Où trouver de l'aide ?

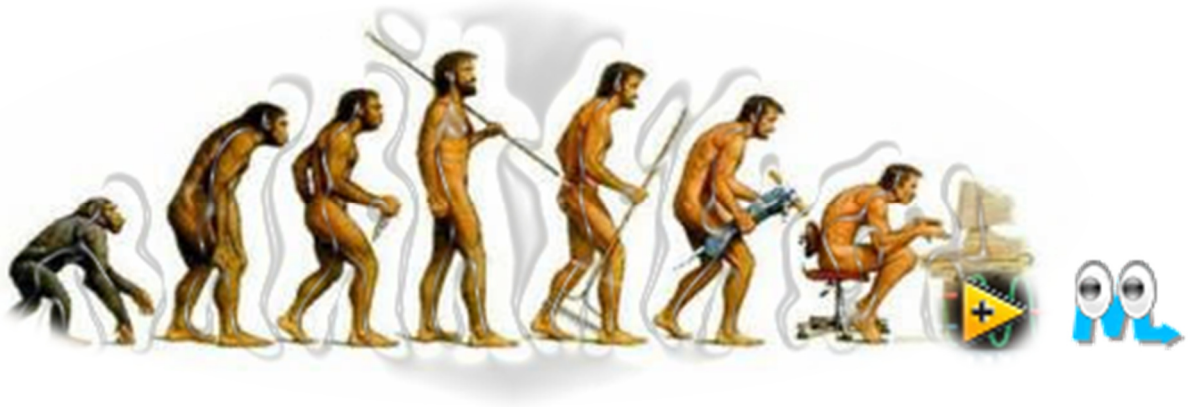
- La communauté travaille pour vous
  - 2 Forums National Instruments Francophone
    - http://forums.ni.com/t5/Discussions-au-sujet-de-NI/bd-p/4171
    - http://forums.ni.com/t5/Discussions-au-sujet-des-autres/bd-p/4170
  - 1 communauté Francophone
    - https://decibel.ni.com/content/community/regional/france/labview
    - Rencontre développeurs : cf LUGE
  - Mais aussi… l'ensemble des forums et communautés de NI
    - **LabVIEW Development Best Practices**
    - Large Applications : ni.com/largeapps

Livre Français

- **Livre LabVIEW : programmation et applications**
  - 3iéme Édition, Dunod 05/2015
  - Lien internet du livre

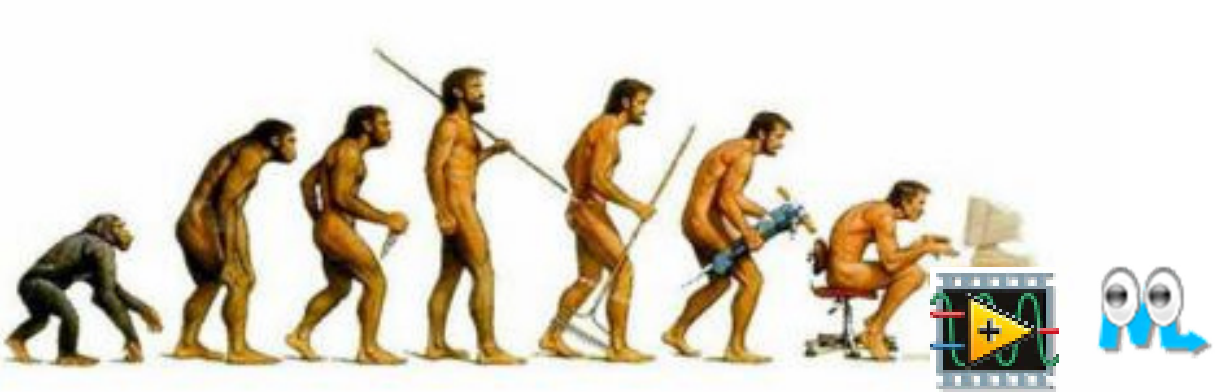
Chapitre 3 du livre = cette présentation (en plus court)

II] Histoire de la présentation



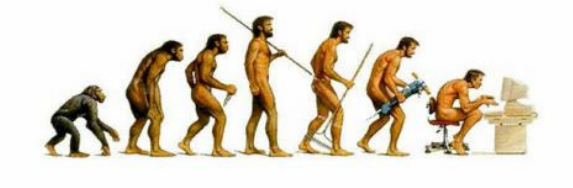
Histoire de Darwin

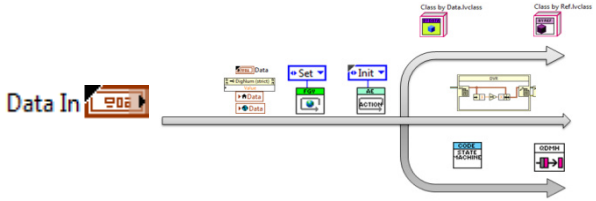
- Charles Robert Darwin (1809 – 1882)
- Hypothèse : tous les êtres vivants ont un seul ancêtre commun
- Les « êtres vivants » ont évolué et se sont adaptés
- Processus connu sous le nom de « sélection naturelle » (la loi du plus fort) ou théorie de l'évolution.



Darwin appliquée à LabVIEW

- Humour
- « Personnellement j'aime » le // entre
  - « être vivant » qui s'adapte (pour survire)
  - La gestion des données sous LabVIEW qui s'adapte (pour éviter « les problèmes »)
- Evoluer pour s'adapter aux besoins

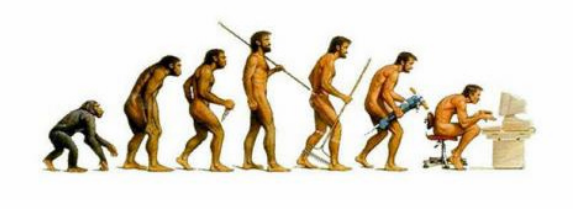


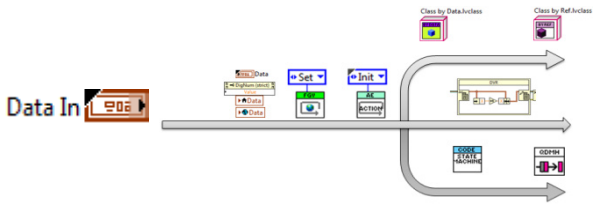


Darwin appliquée à LabVIEW

- Comment et pourquoi la Mémorisation des données a-t-elle évoluée?

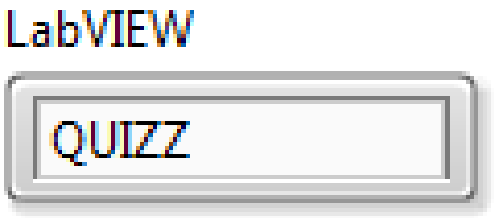
Darwin	LabVIEW
Un seul ancêtre commun	Contrôle (le)
à tous les êtres vivants	À toutes les techniques de mémorisation des Données
S'adapter pour Survivre	S'adapter pour Eviter les bugs
La loi du plus fort	Ajouter des fonctionnalités





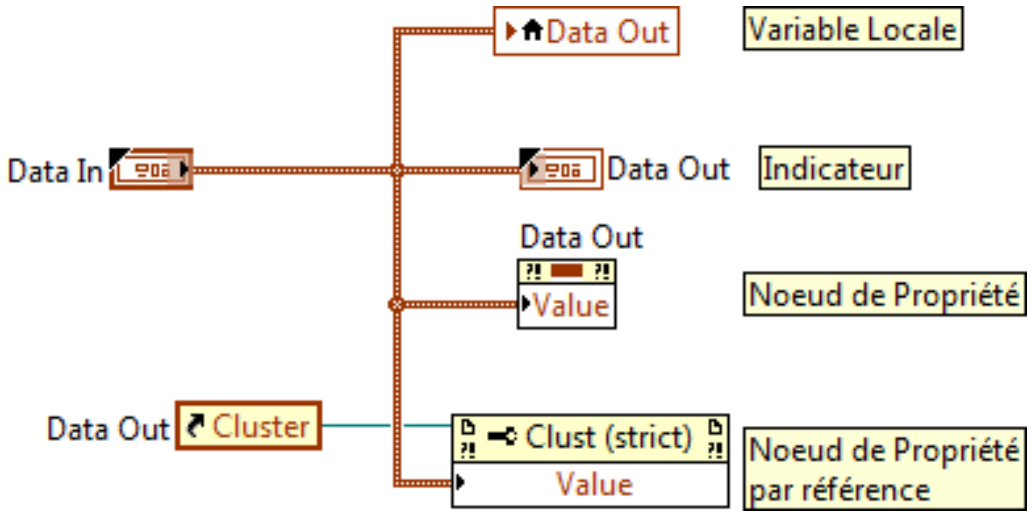
- Il était une fois…. Une donnée

II] Evolution famille « données »



Quizz N°1

- **Pour transmettre la donnée « Data In », faut-il mieux connecter le contrôle sur :**
  - A. La variable locale de « Data Out »
  - B. Le terminal de l'indicateur « Data Out »
  - C. Le nœud de propriété implicite « value » de « Data Out »
  - D. Le nœud de propriété par référence « value » de « Data Out »



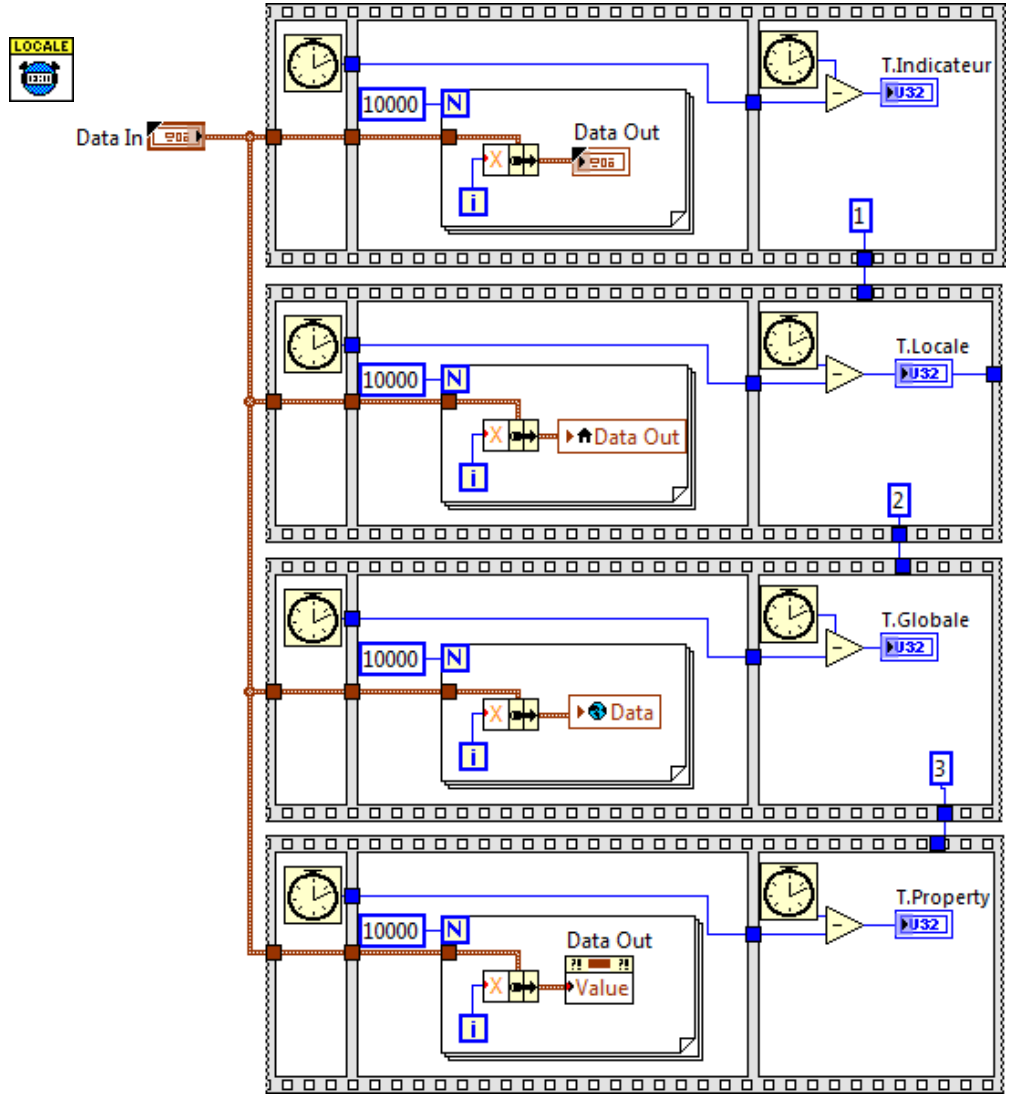
Expliquer la réponse du Quizz N°1

L'explication de la réponse n'est pas si simple

- **Comme démontrer $x \times 0 = 0$ … Parce que**
- **Avant de donner la véritable réponse :**
  - Vous faire sentir (la réponse) par une mesure
  - Vous pouvez me croire… ou pas

Expliquer la réponse du Quizz N°1

Démonstration



Expliquer la réponse du Quizz N°1

Explication « simplifiée »

- **Définition Les nœuds de propriété :**
  - Se synchronisent sur le thread « interface utilisateur ». Seul système d'exécution pas multithread (Bloquant)
  - Forcent à redessiner la face-avant à chaque appel (Lenteur)
  - Forcent la face-avant à être chargée en mémoire (les commandes et indicateurs auront plusieurs copies de données supplémentaires – diagramme et face-avant)

Résultat Quizz N°1

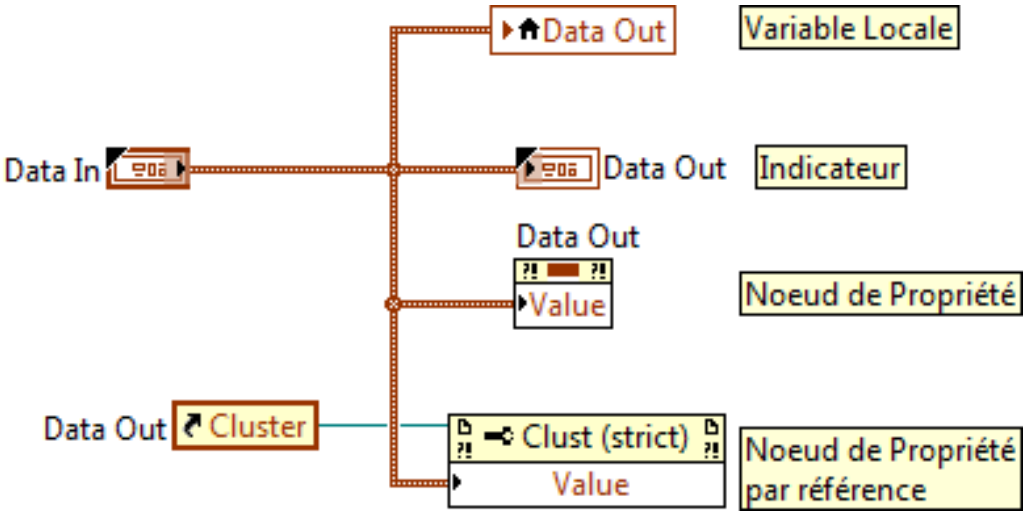
- **Nœud de propriété**
  - Copies donnée
  - Que Thread UI
  - Lenteur, bloquant
  - Force IHM en mémoire
  - Race Condition

V.S.

- **Locale**
  - Copies donnée
  - Tout les Threads
  - Race condition

Réponse Quizz N°1

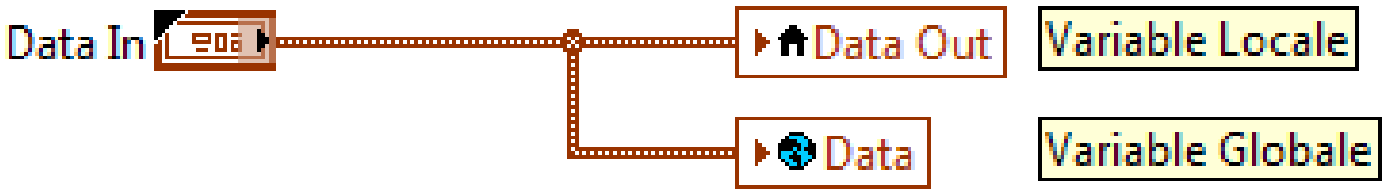
1. L'indicateur « Data Out »
2. La variable locale (copie buffer)
3. Le nœud de propriété implicite (thread UI + copie buffer)
4. Le nœud de propriété par référence (référence)



Eviter 3. et 4. juste pour « value »

Quizz N°2

- Pour transmettre la donnée, faut-il mieux connecter le contrôle « Data In » sur :
  A. La variable locale de « Data Out »
  B. La globale « Data »



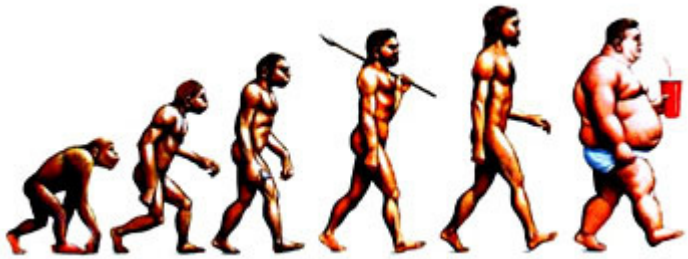
Résultat Quizz N°2

Match Nul(presque)

A éviter

- **Eviter les variables Locale – Globale - nœud de propriété**
  - Copies des données (attention « Grosse Structure »)
  - Lenteurs
  - Pas de protection contre « Race Conditions »
  - Pas de gestion d'erreur

- **1 fois (OK) mais L'exagération nuit à l'évolution**



A éviter

- Eviter les variables Locale – Globale - nœud de propriété

Comment ????

FGV : Functional Global Variable

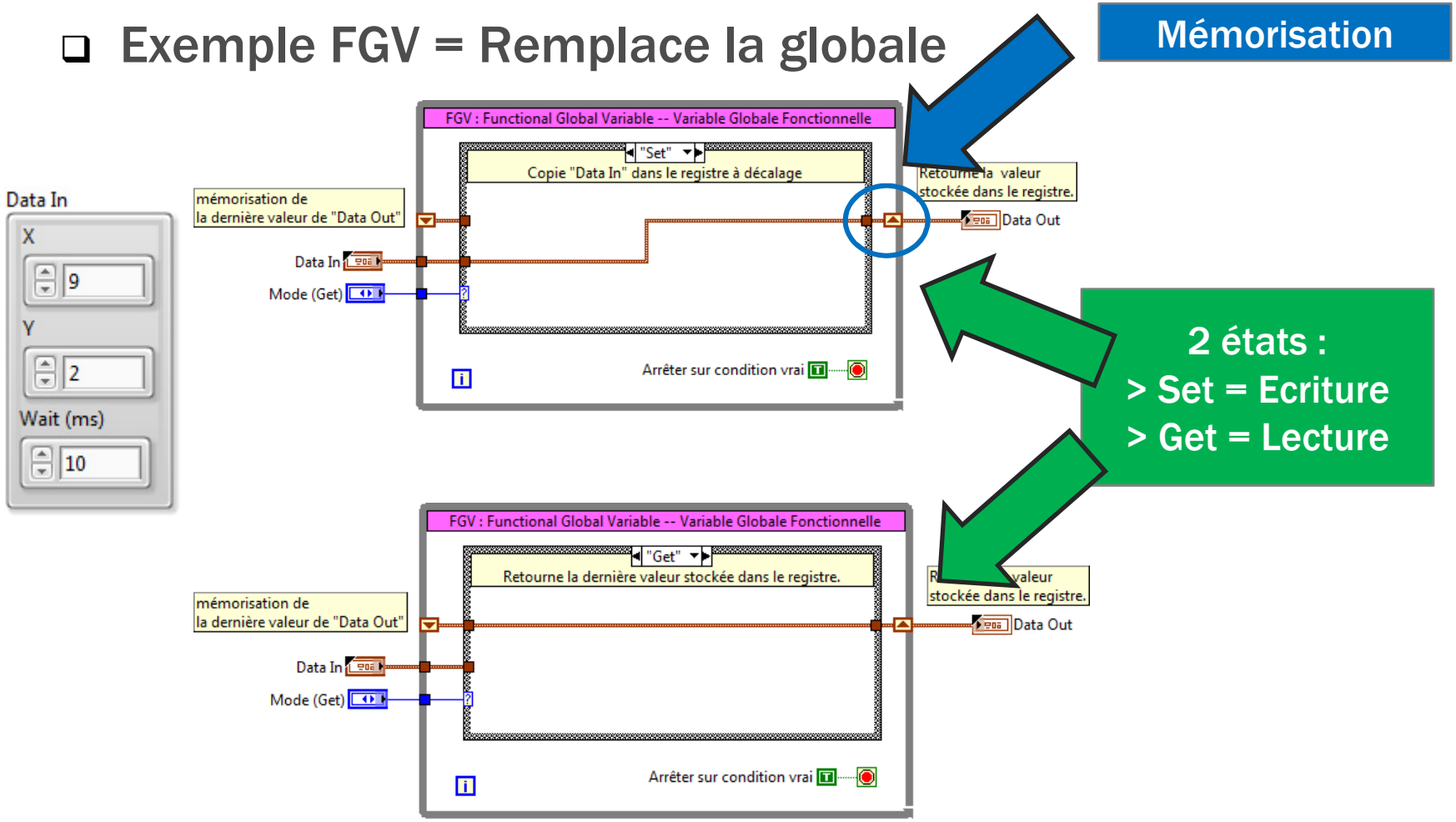
- **FGV : Functional Global Variable ou LV2**
  - Registre à décalage non initialisé, d'un vi non réentrant
  - Mémorise une valeur sur un Set (Write)
  - Retourne la valeur sur un Get (Read)

Quizz N°3

- Vous connaissez tous?
- Pourquoi le nom FGV ou variable LV2?

Exemple FGV

- Exemple FGV = Remplace la globale

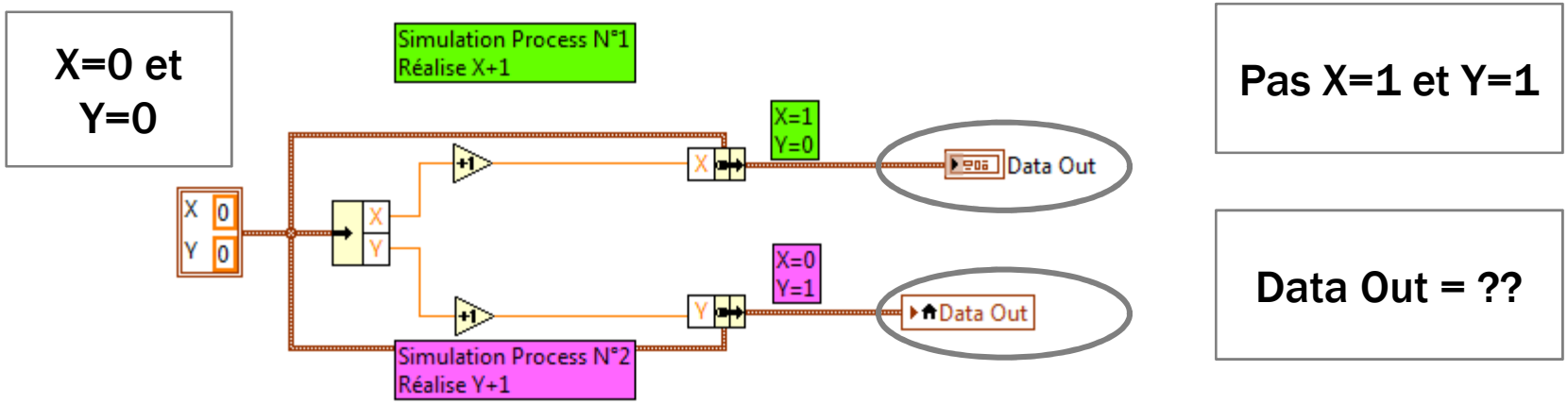


Race condition

- Situation de compétition ?????????????

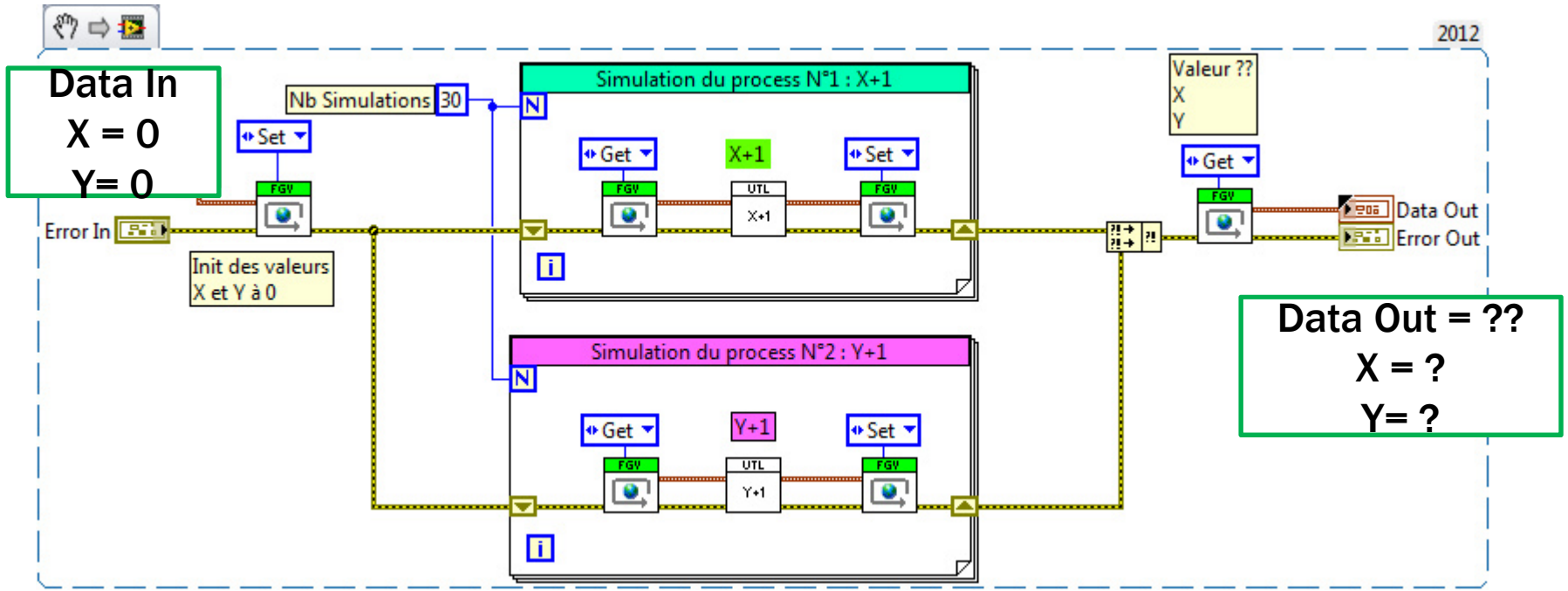
Rappel « bug » Race condition

- Situation de compétition
- « Bug » Accès « concurrent » de 2 codes à la même variable.
- Valeur « instable » de la variable



Quizz N°4

- La FGV évite-t-elle « le bug » des accès concurrents – Race Conditions?



Résultat Quizz N°4

- Non

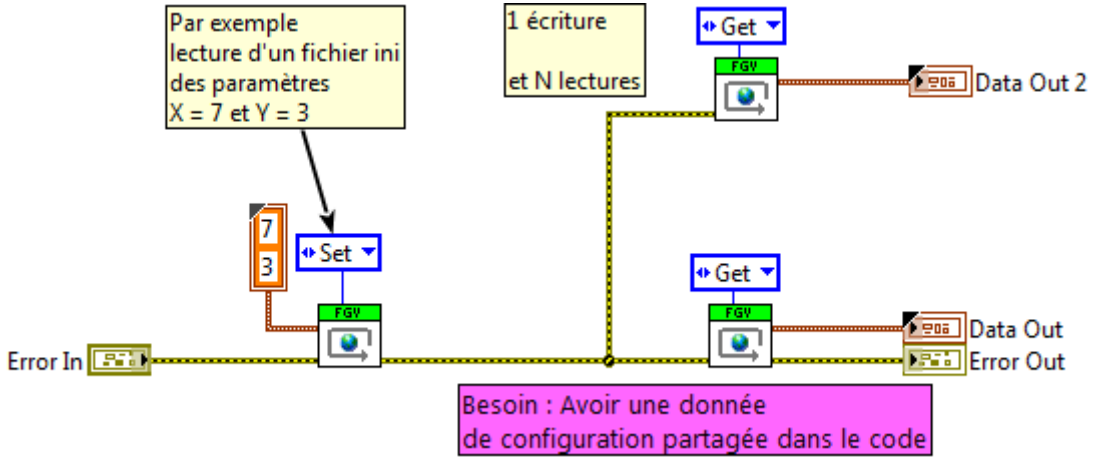
Démonstration code LabVIEW

Code téléchargeable sur le site MESULOG >> Société >> Présentations Techniques

www.mesulog.fr/presentations_techniques

Utilisation recommandée de la FGV

- 1 écriture
- N Lectures

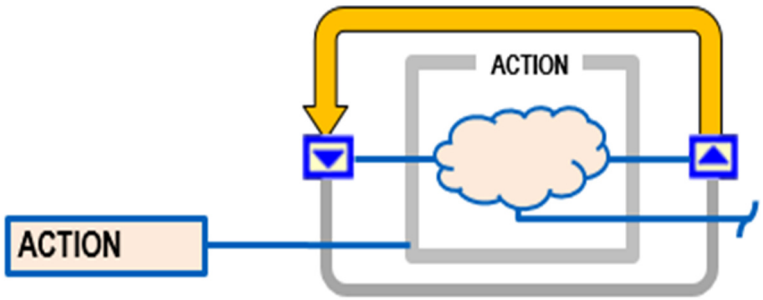


Race condition

- Comment éviter les Situations de compétition ?????????????
  - N écritures
  - N Lectures

AE : Action Engine

- **AE : Action Engine**
  - Evolution de la FGV en ajoutant des actions,
  - Principe de l'encapsulation
  - La donnée reste dans la fonction logicielle = protection

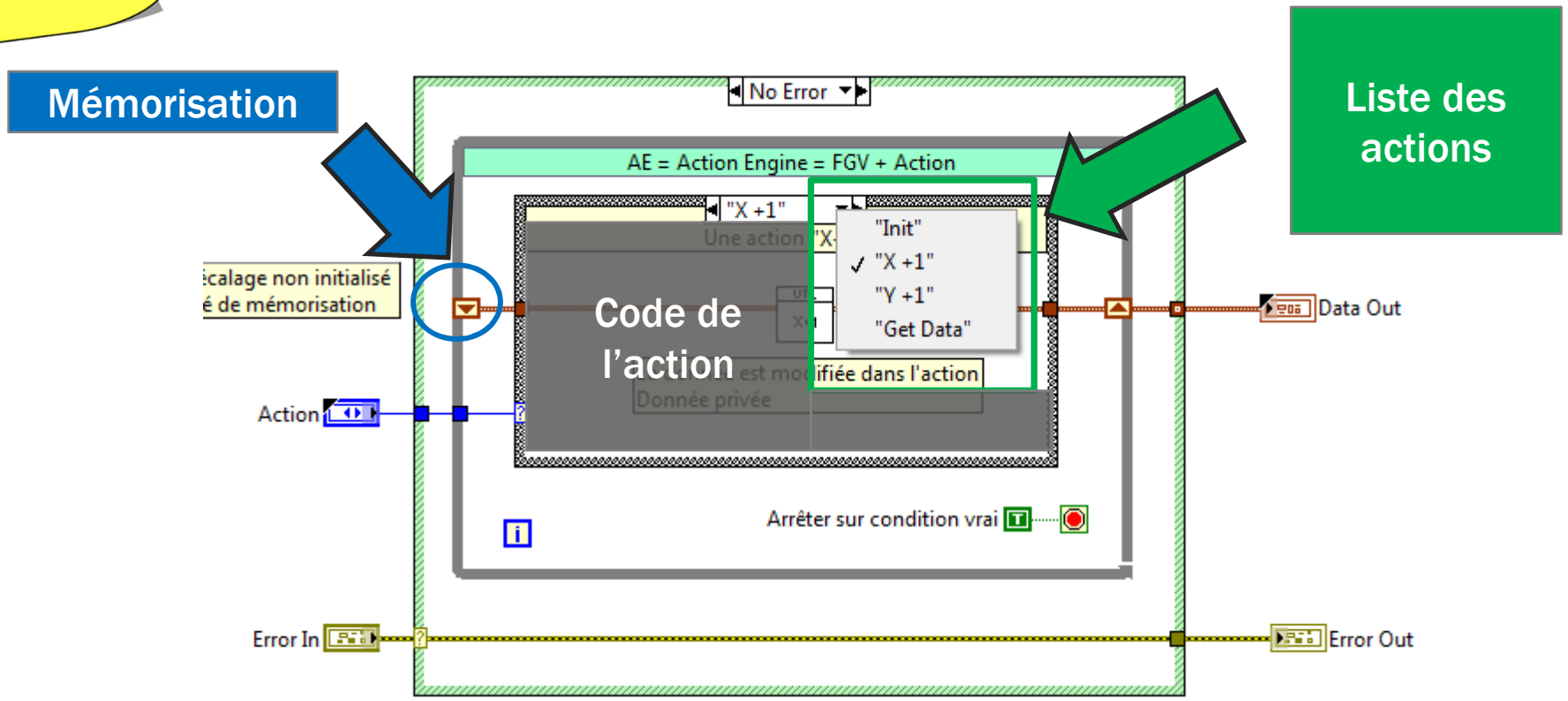


Exemple AE

Exemple AE

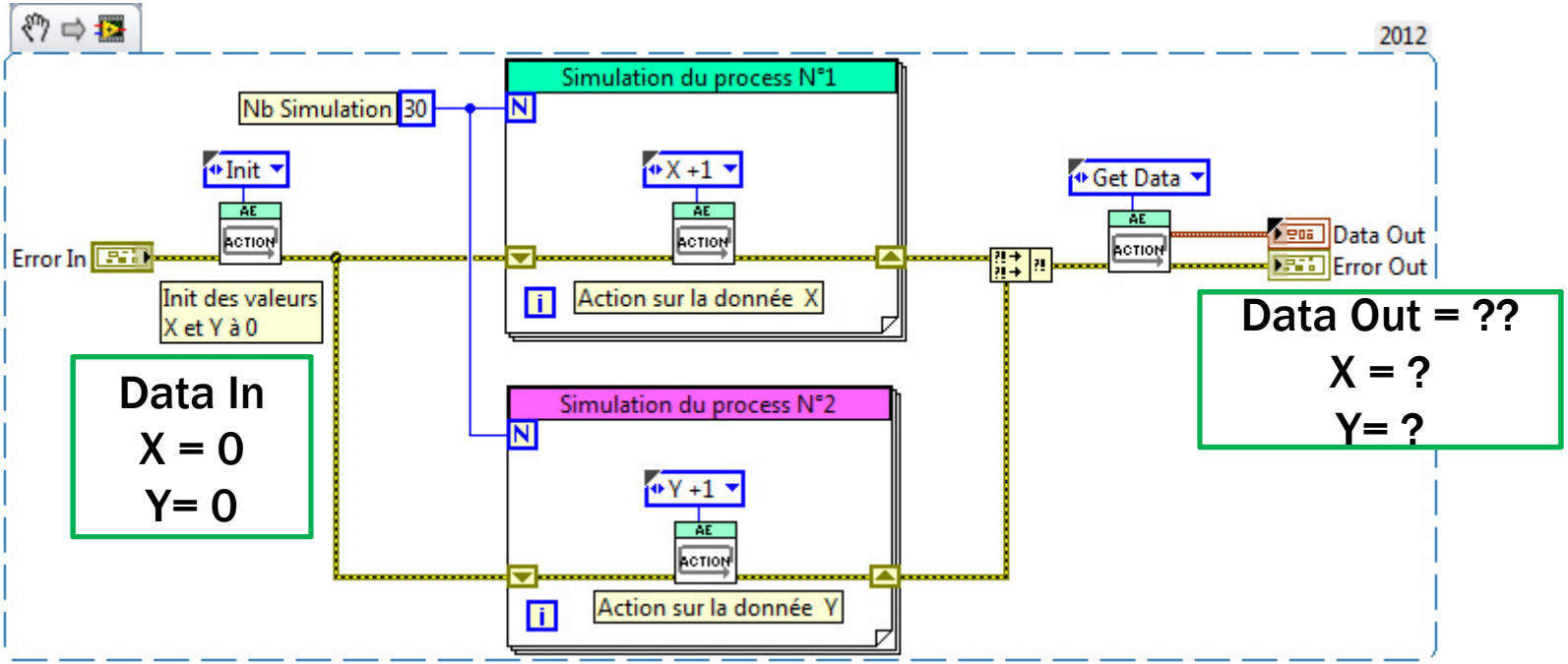
Important Message

- FGV + Actions
- Code autonome – réutilisable – Gestion erreur



Quizz N°5

- L'AE évite-t-elle « le bug » des accès concurrents – Race Conditions?



Résultat Quizz N°5

- Oui, via l'encapsulation : rassembler les données et les méthodes au sein d'une même structure

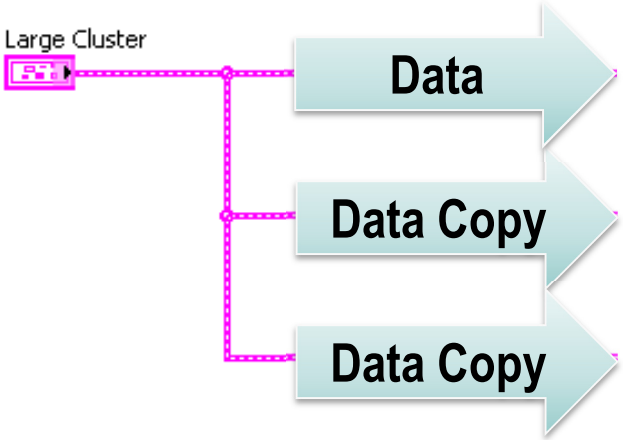
Démonstration code LabVIEW

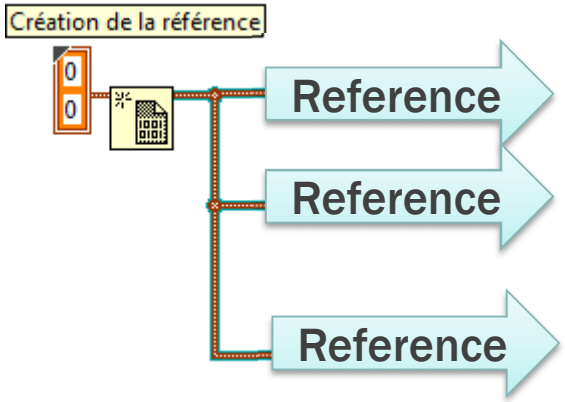
Code téléchargeable sur le site MESULOG >> Société >> Présentations Techniques

www.mesulog.fr/presentations_techniques

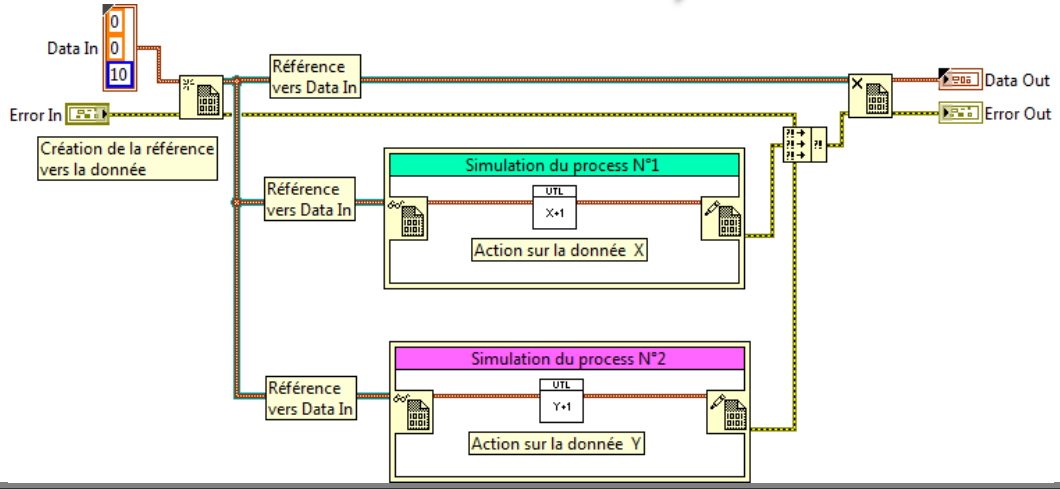
DVR : Data Value Reference

- Manipulation de la référence de la donnée mais pas la données : pas de copie, *pointeur zone mémoire*





- Protection contre les accès concurrent



Quizz N°5

- **La DVR (Data Value Reference) permet-elle? :**
  A. D'éviter les copies de données (« grosse structure »)
  B. De protéger contre les accès concurrents – Race Conditions
  C. De gérer les erreurs

Résultat Quizz N°5

- Réponse A, B et C : Oui

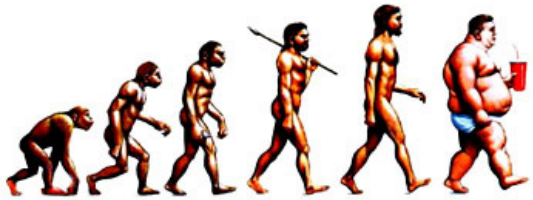
Démonstration code LabVIEW

Code téléchargeable sur le site MESULOG >> Société >> Présentations Techniques

www.mesulog.fr/presentations_techniques

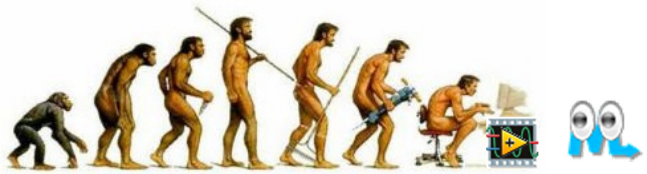
Synthèse évolution « données »

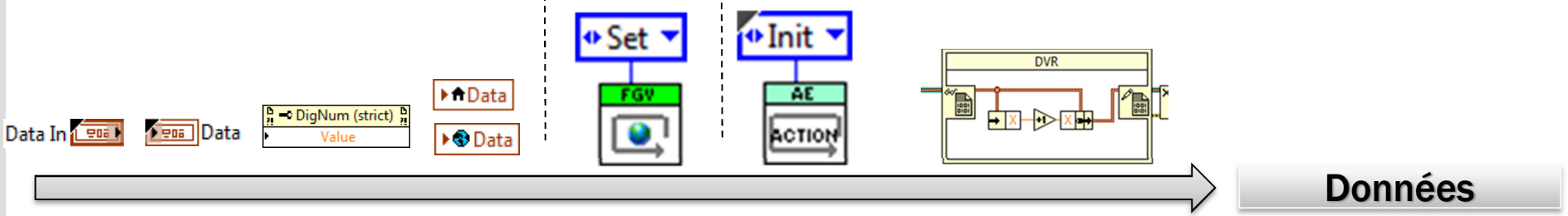
1 fois (OK) mais
L'exagération nuit à l'évolution



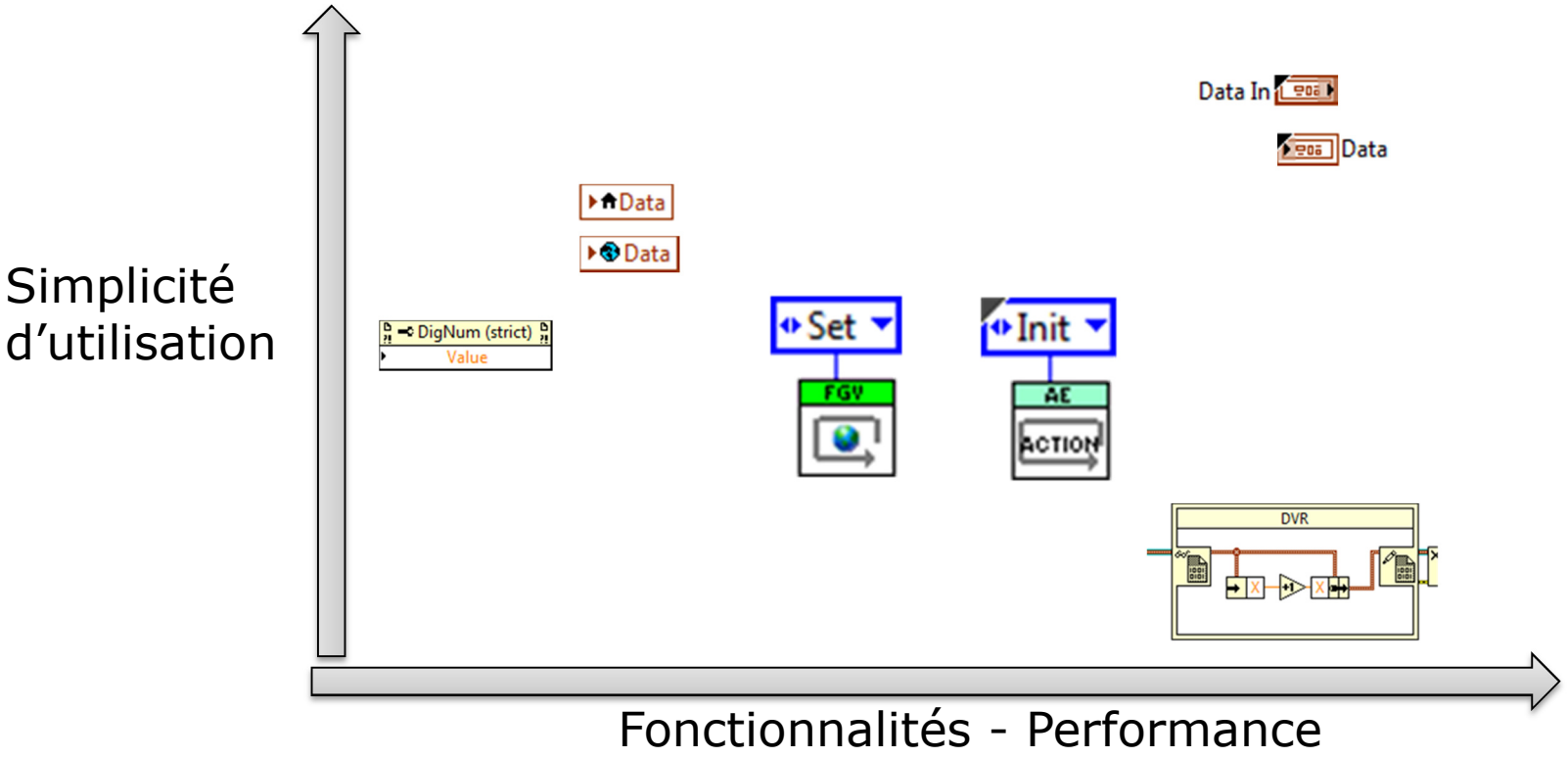
S'adapter à un besoin

- N écritures
- Code autonome
- Réutilisable
- Gestion erreur personnalisée





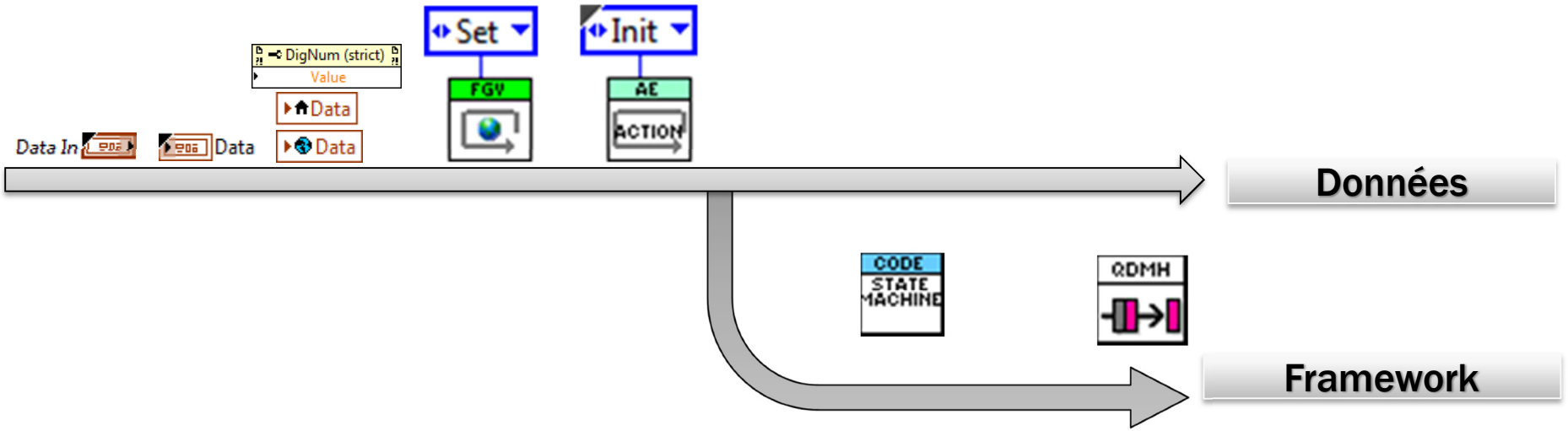
Synthèse évolution « données »



Évolution structure application

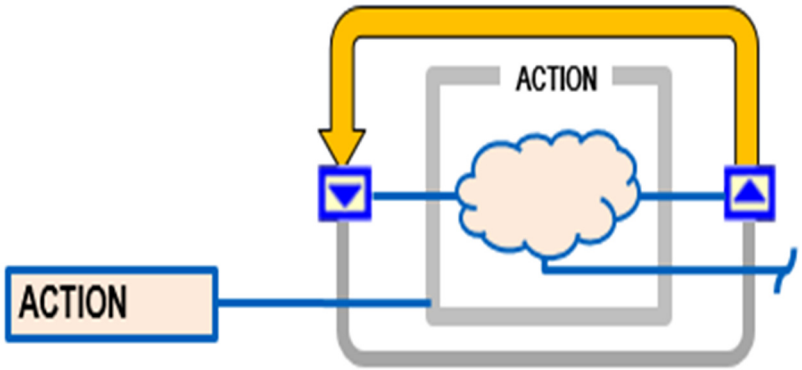
- Toutes les techniques permettent de gérer des données
- Pas des applications

- Evolution techniques pour gérer les données vers une structure pour gérer les applications



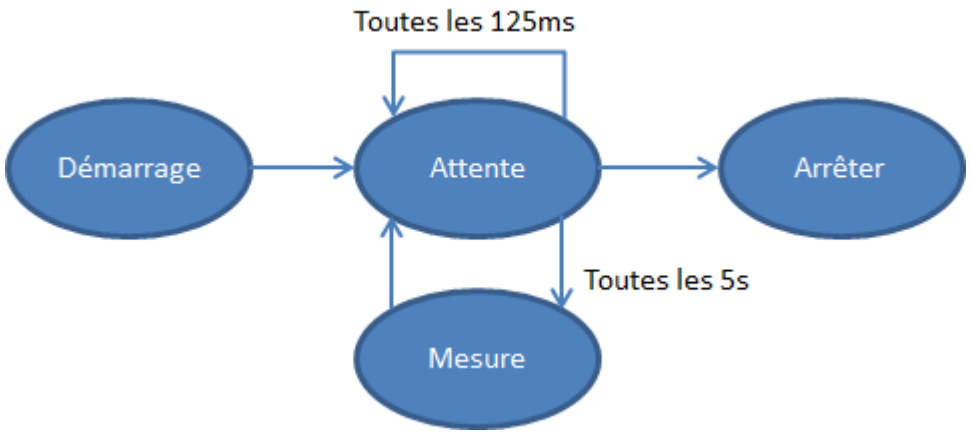
De l'action à l'état

- **Rappel l'Action Engine (AE) :**
  - Réalise une action sur la données (exemple X+1 ou Y+1)
  - Puis fin
  - Jusqu'au prochain appel de la fonction logicielle



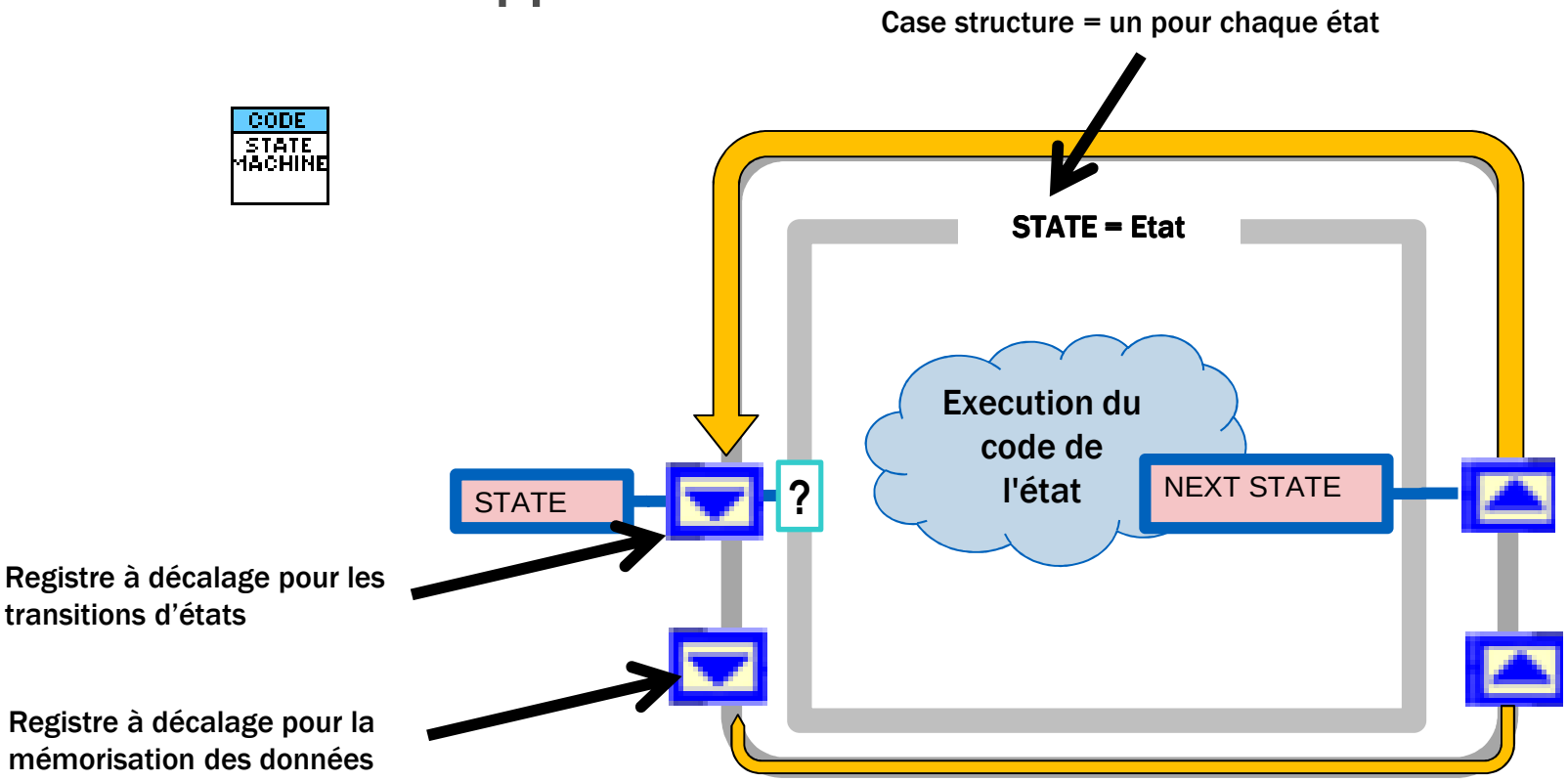
Pour une application : Machine à états

- **Modèle « Machine à états », basé sur diagramme états-transitions:**
  - Etats = actions à réaliser
  - Chaque section de code détermine la transition suivante
  - Découpage clair des tâches à effectuer
  - Ne « sort » du code que si fin du programme principal



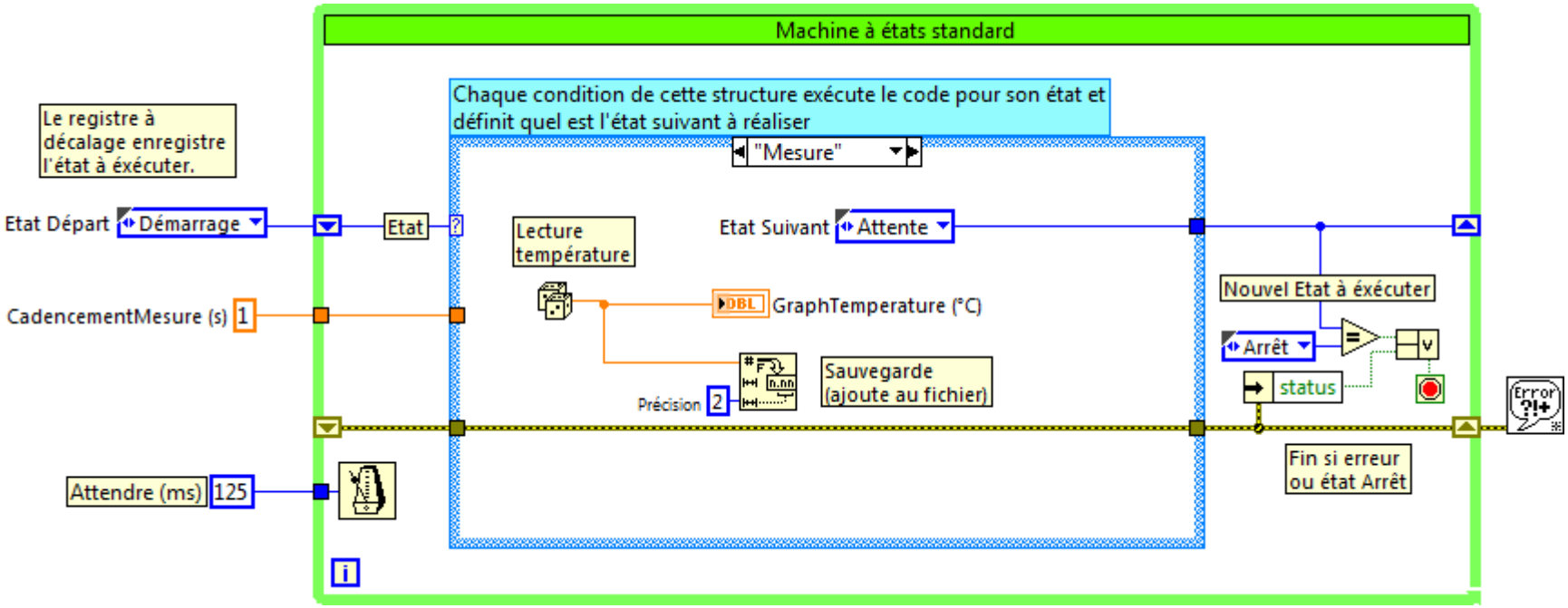
- Tout le monde connait cette technique avec LabVIEW?

- La Machine à états, ou State Machine (SM), est une structure d'application



Exemple code LabVIEW de la SM

- *Démonstration*



Synthèse de la Machine à états

- Fondamentale du développement logiciel LabVIEW
- Code simple, documenté et intuitif
- Point faible : peut perdre des états si en même temps
- Idées d'évolutions ?

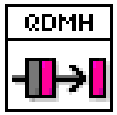
Autres évolutions « Framework »

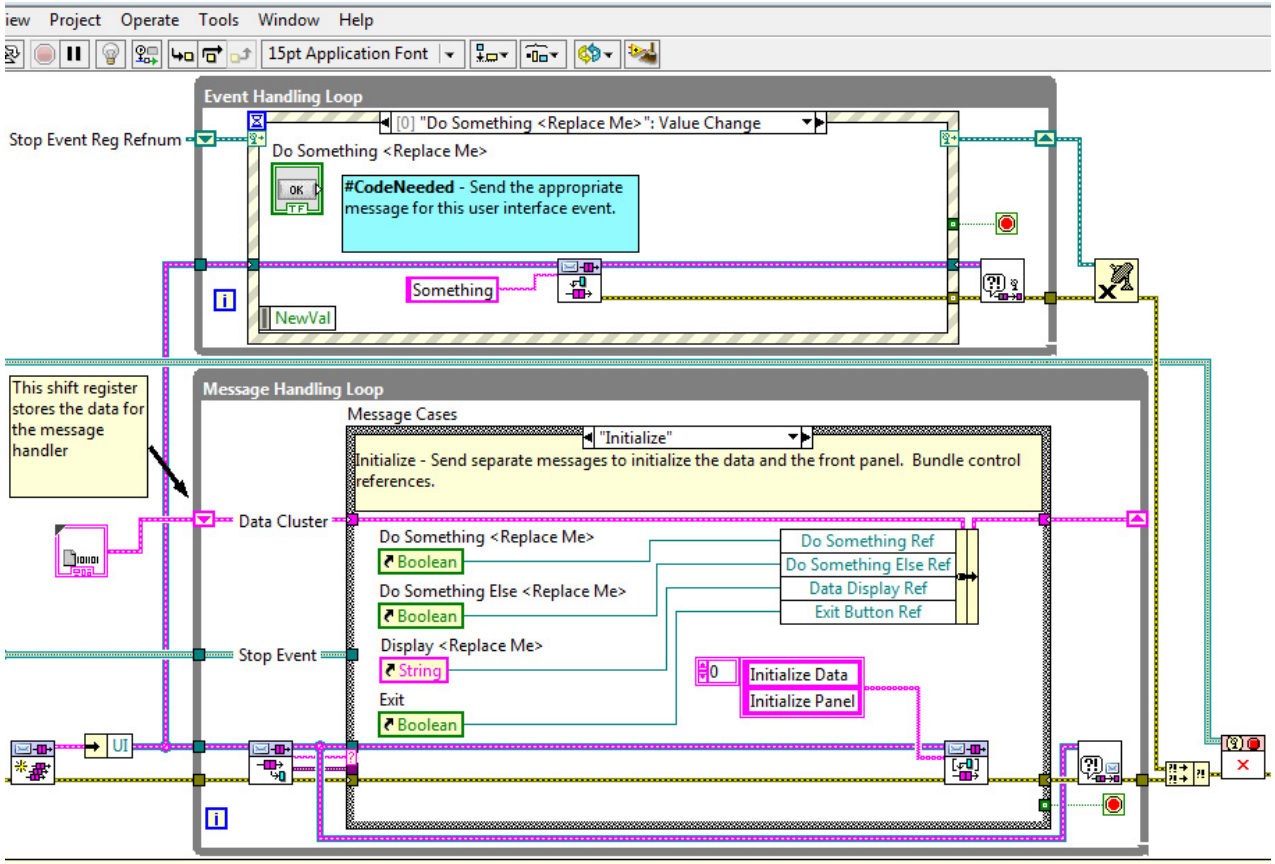
- **Quelques exemples disponibles**
  - Producteur/consommateur avec File d'attente (Queue)
  - Boite de dialogue avec structure évènementielle (event driven)
  - *Gestionnaire de Messages dans une File d'attente (GMF en Français ou QDMH en English) = INDISPENSABLE*
  - Actor Framework

- **En Annexe : QDMH : indispensable pour vos projets**

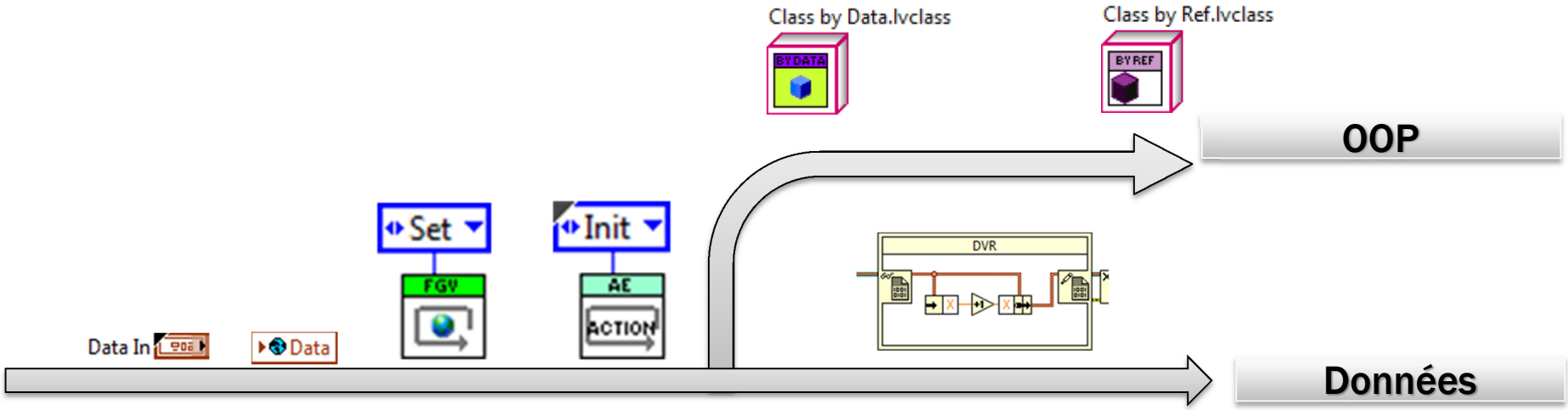
Le modèle de projet QDMH

- Queue Driven Message Handler (QDMH)
- LabVIEW 2012 : les modèles de projet LabVIEW via le gestionnaire de projet.



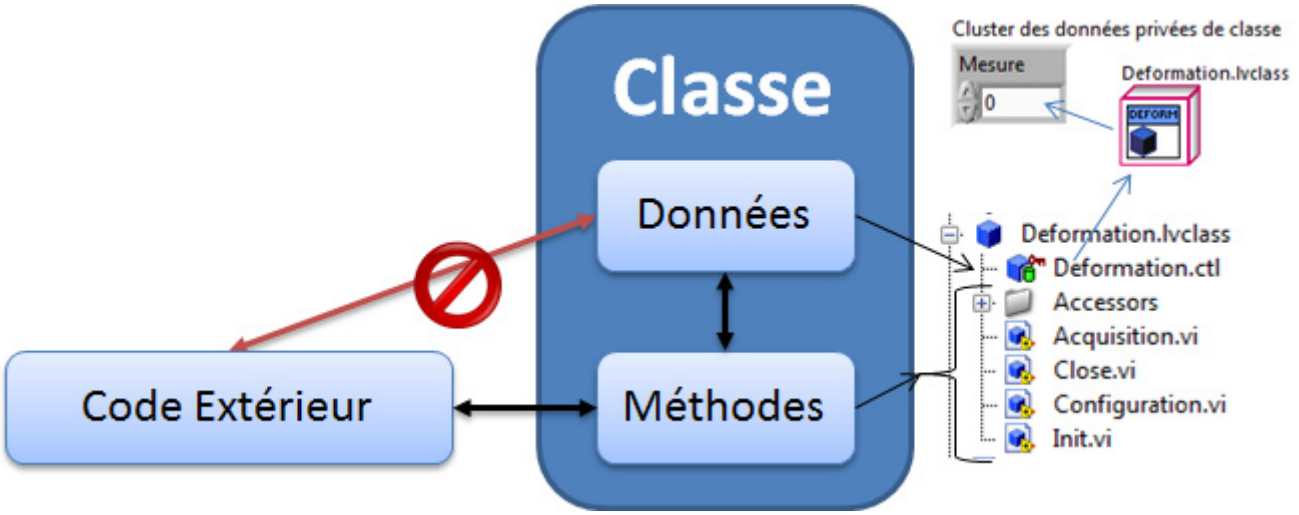


- Evolution de la gestion des données vers OOP



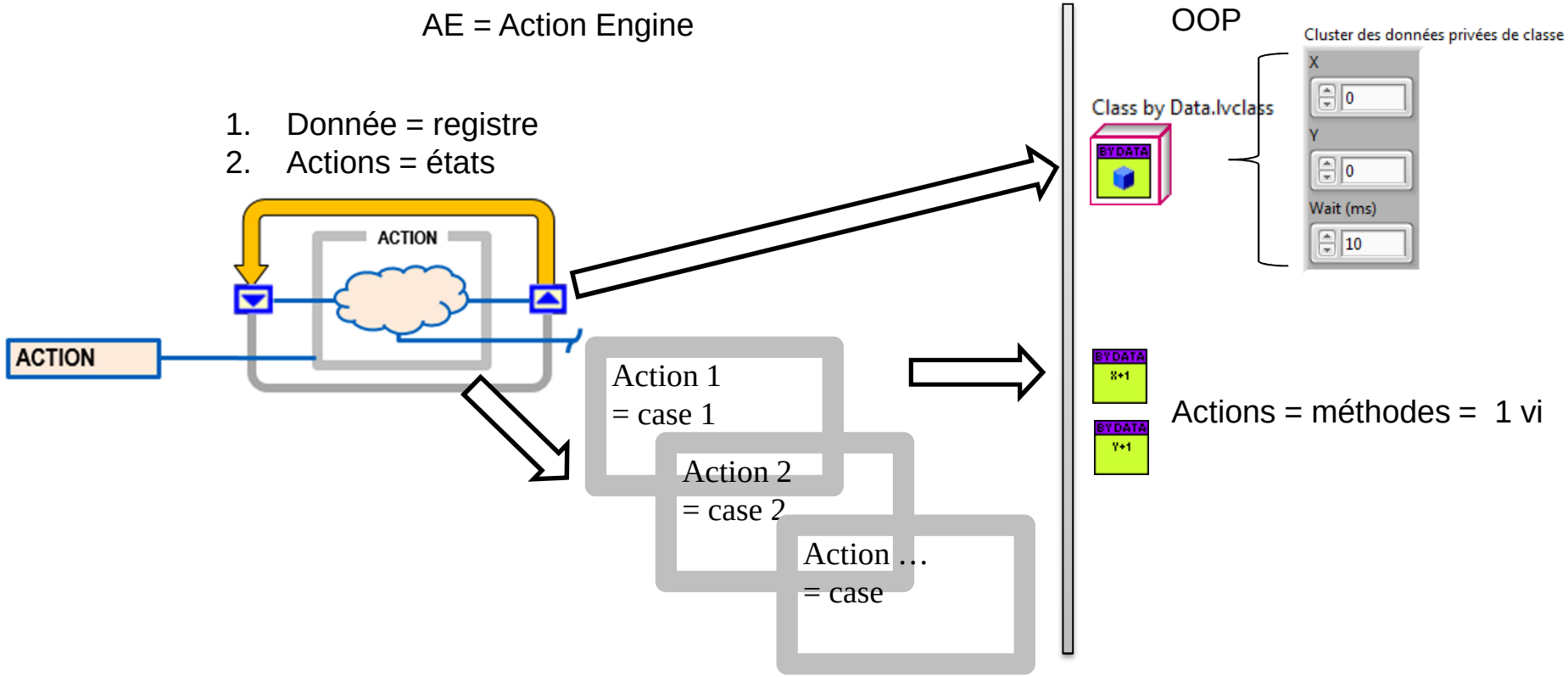
Programmation Objet sous LabVIEW

- Une classe est un ensemble de données (privées) et d'actions (méthodes) qui permettent d'agir sur ces données
- + Concept : héritage, dispatch dynamique
- + Concept : encapsulation



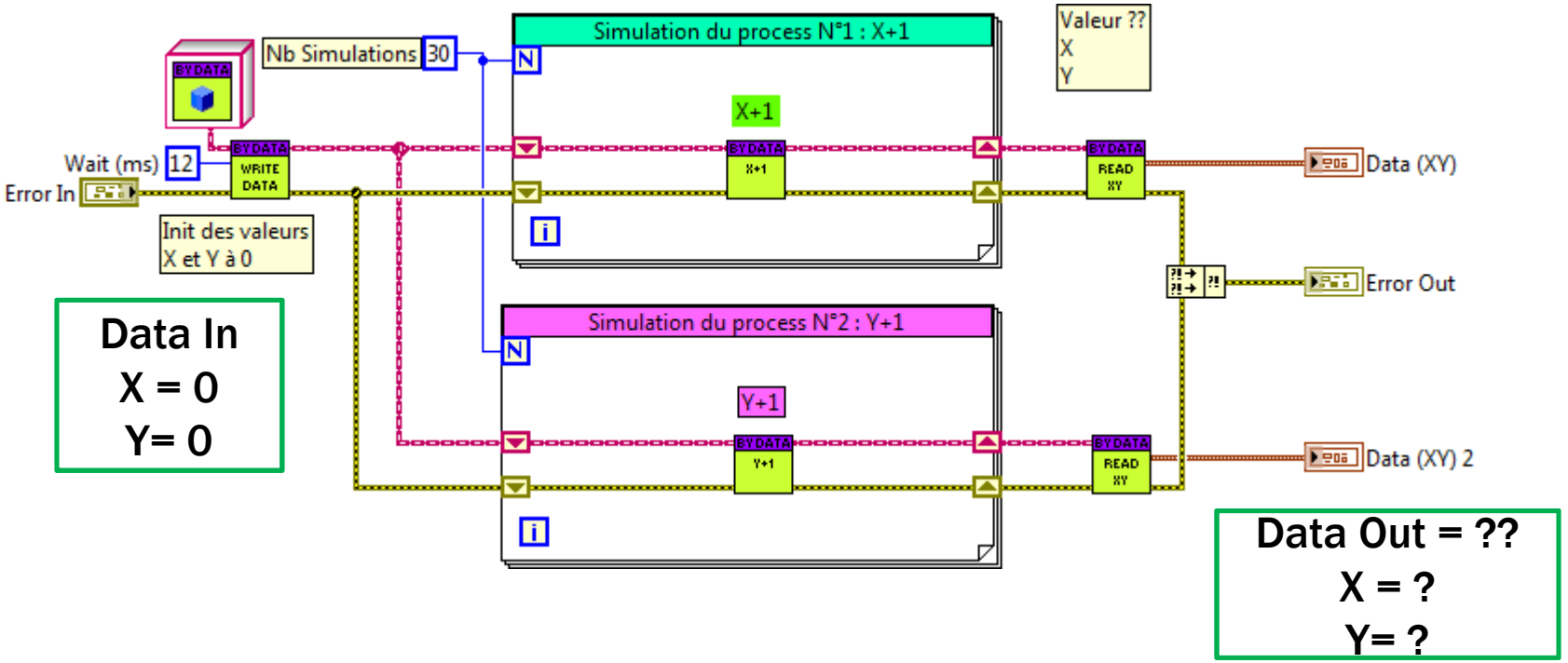
OOP évolution de l'ancêtre AE (Action Engine)

- Registre à décalage devient données de la class
- Actions devient Méthodes (encapsulation)



Quizz N°6

- OOP évite-t-elle « le bug » des accès concurrents – Race Conditions?



Résultat Quizz N°6

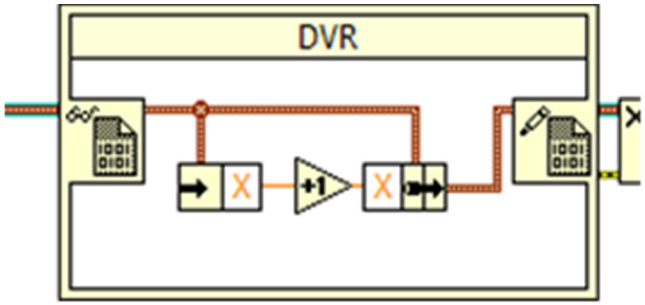
- Non (by data)

Démonstration code LabVIEW

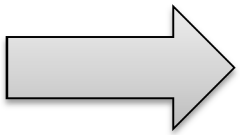
Code téléchargeable sur le site MESULOG >> Société >> Présentations Techniques

www.mesulog.fr/presentations_techniques

Objet by value vers by Ref



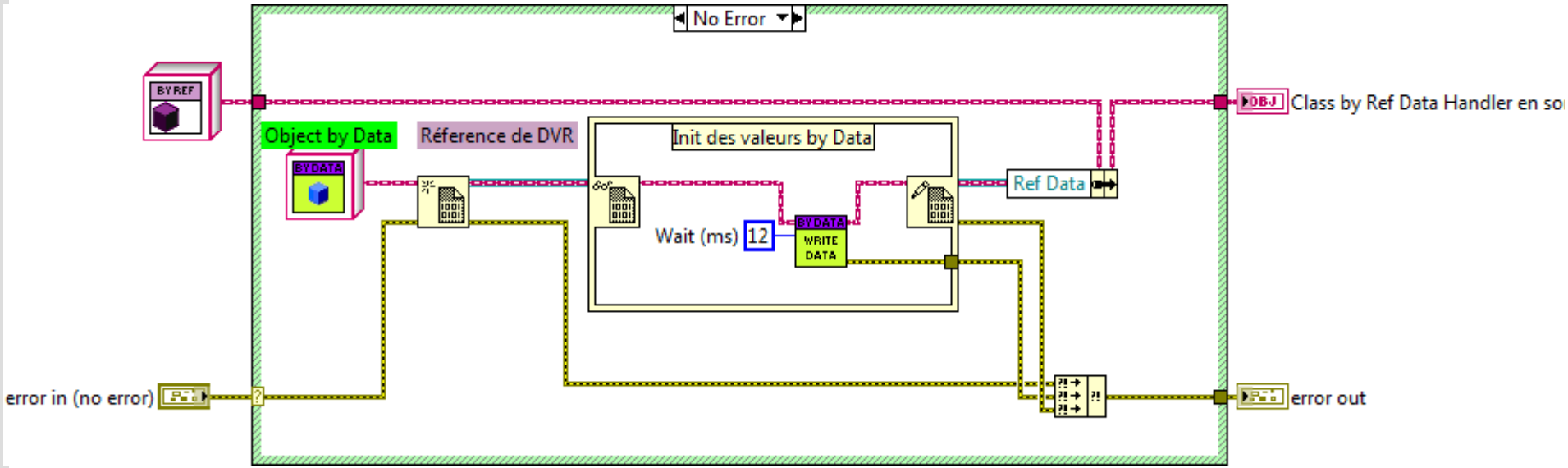
Class by Data.lvclass



Class by Ref.lvclass

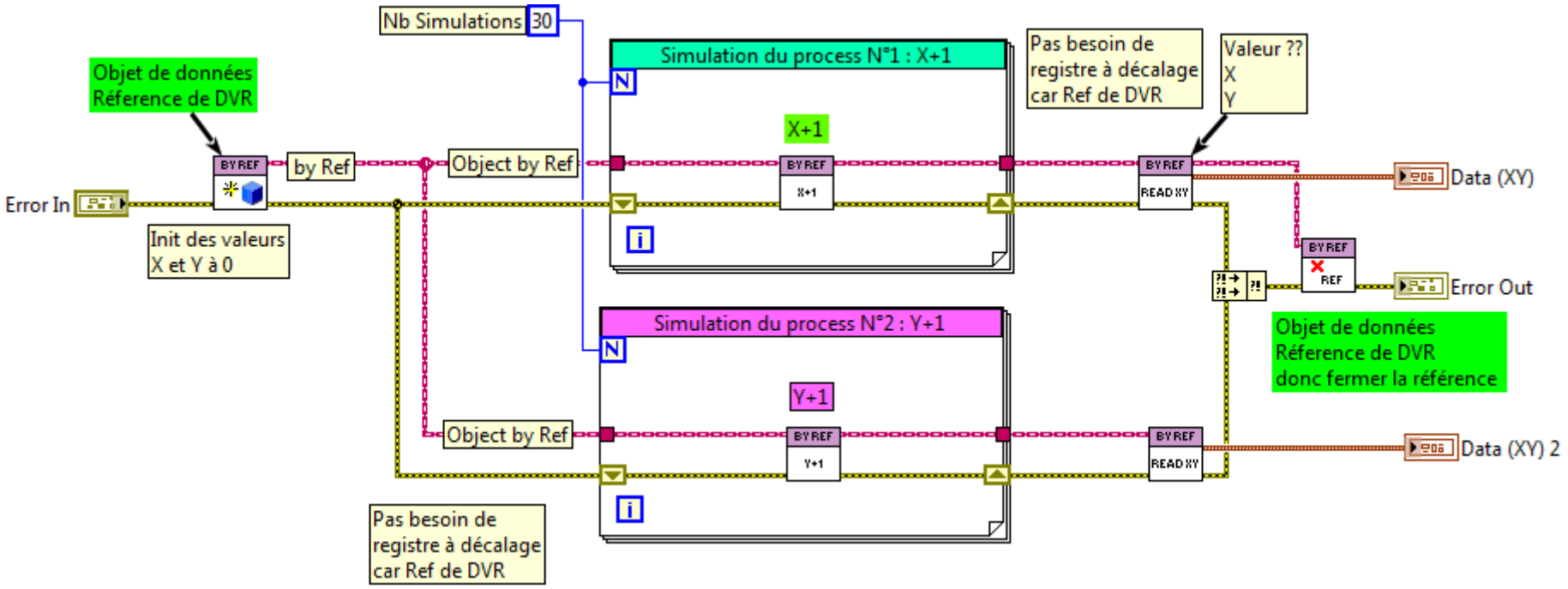
Objet by value vers by Ref

- Remarque : par défaut by Value
- Possibilité : Utiliser la DVR pour créer une référence
  - DVR + OOP by data = by Ref



Quizz N°7

- OOP by Ref évite-t-elle « le bug » des accès concurrents – Race Conditions?



Résultat Quizz N°7

- Oui (by ref)

Démonstration code LabVIEW

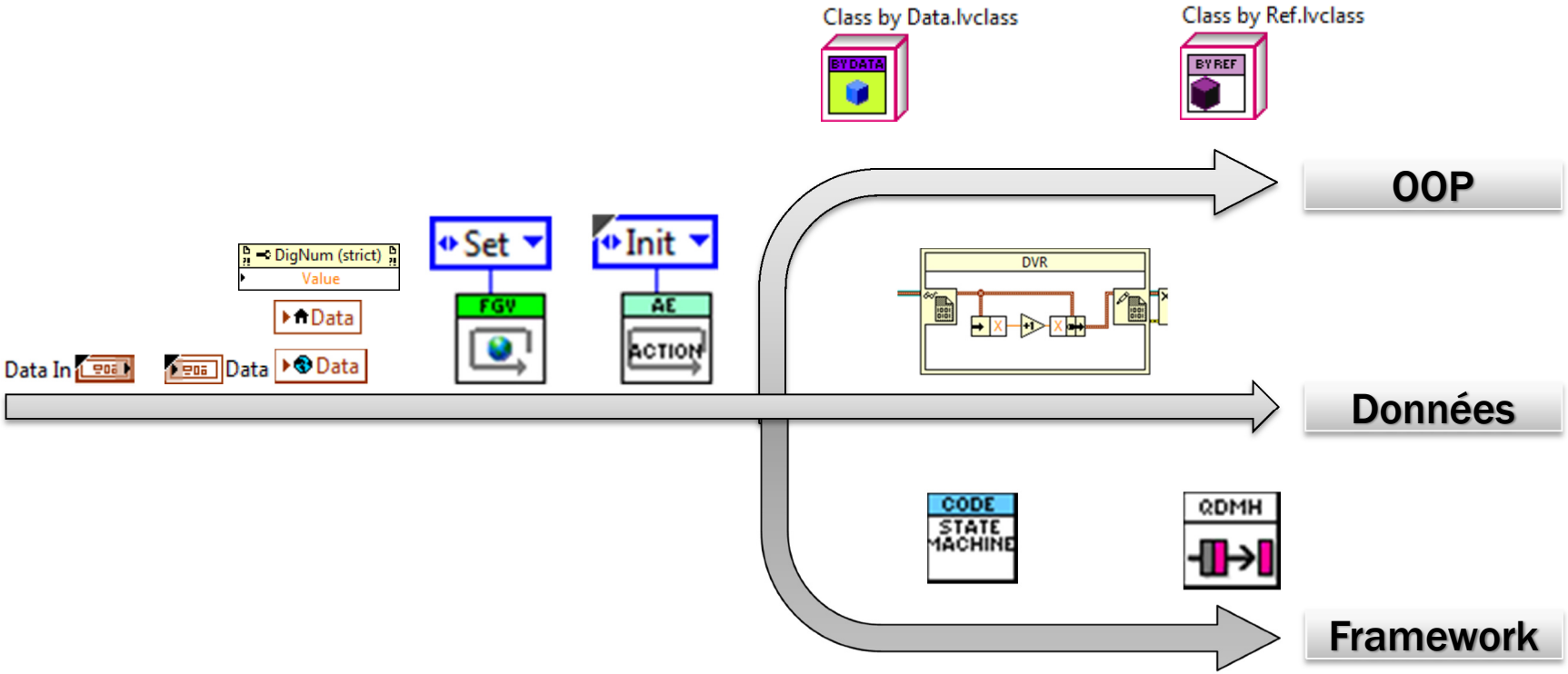
Code téléchargeable sur le site MESULOG >> Société >> Présentations Techniques

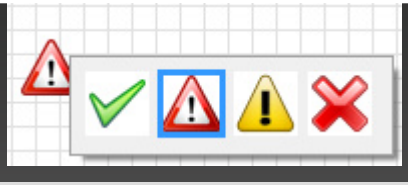
www.mesulog.fr/presentations_techniques

Conclusion : évolution pour LabVIEW

L'exagération nuit à l'évolution

S'adapter à un besoin pour éviter les bugs et ajouter des fonctionnalités





Liens

Plus de présentations techniques

- www.mesulog.fr/presentations_techniques
- National Instruments : luc desruelle's Blogue
- Tutoriels developpez.com : luc Desruelle

Plus de livres

"LabVIEW programmation et applications" 3iéme édition, Dunod

Plus de National Instruments Francophone

- Forum francophone NI LabVIEW
- Forum francophone Autres produits NI
- Communauté Francophone

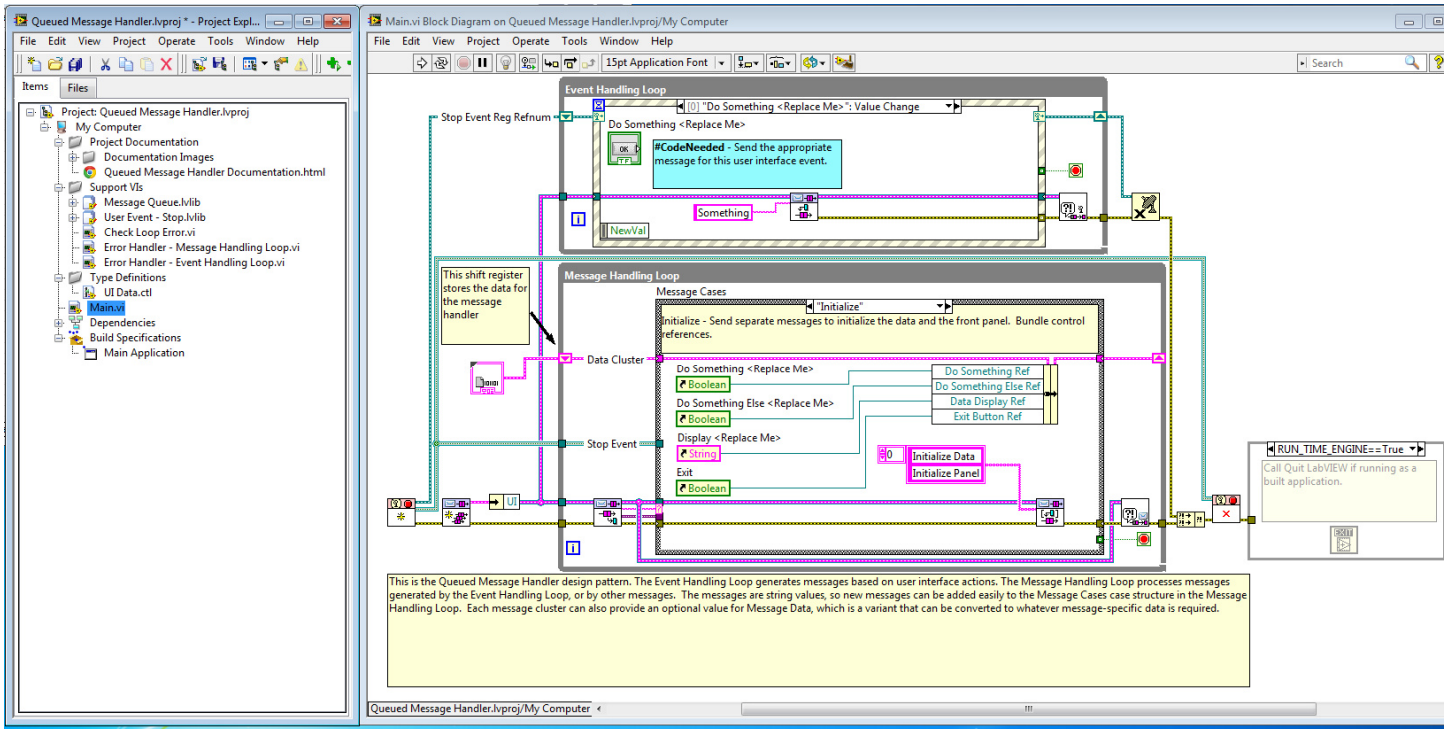
Luc Desruelle | LinkedIn

Questions

- Annexe 1
- Le modèle de projet QDMH

Le modèle de projet QDMH

- Queue Driven Message Handler (QDMH)
- LabVIEW 2012 : les modèles de projet LabVIEW via le gestionnaire de projet.



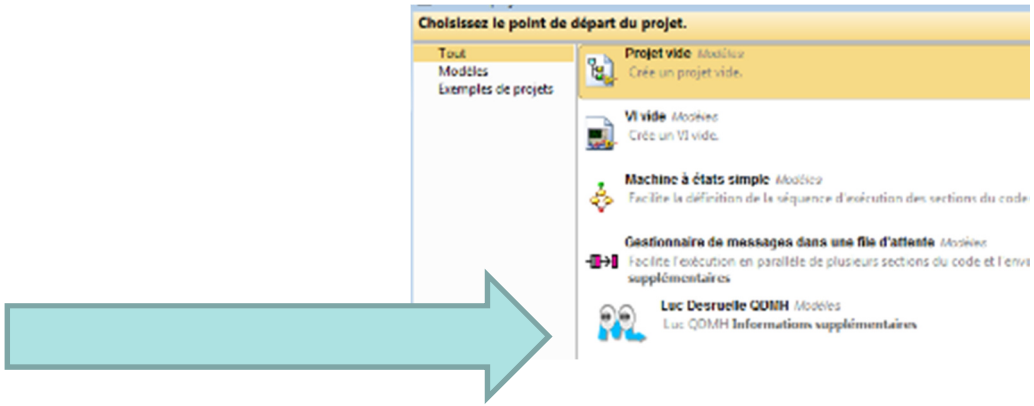
Le modèle de projet QDMH

- La structure repose sur un modèle producteur – consommateur
  - ***la boucle productrice :*** la structure évènementielle
    - capture les actions utilisateurs, sur la face-avant, et produit le « message » via une FIFO
    - Le message est un cluster composé d'un état « case » et une donnée facultative Data de type variant
  - **la boucle consommatrice :** basée sur un modèle de machine à états, dépile sur apparition les données de la FIFO.

Améliorer le QDMH

- **Gestion erreur : quitte l'application sur « erreur »**
  - Remplacer « Exit » par « Error »
  - Ajouter un état « Error » dans la "Boucle de gestion de messages".
    - Affichage de l'erreur
    - Sauvegarde
    - Filtrage si l'utilisateur le décide.
- **N'affiche pas la version du logiciel**
  - Ajouter les VIs de gestion de version de LabVIEW
- **La structure event driven sort sur erreur –**
  - Remplacer par un vi qui transfère l'erreur à la "Boucle de gestion de messages" via la FIFO
- **Supprimer le code "exemple«**
- **...**

- Distribuer votre modèle
- Partie 2/3 : Distribuer des modèles de projet personnalisés avec le gestionnaire de projet LabVIEW - version Simple
- Partie 3/3 : Distribuer son Framework de projet, modèle de projet personnalisé, avec le gestionnaire de projet LabVIEW - version distribution personnalisée



- Annexe 2
- IHM est en mémoire

Mise à jour d'une IHM en mémoire

- **Si l'IHM est en mémoire :**
  - les commandes et indicateurs – sur la face-avant - ont leur propre copie des données