



Modèles d'architecture pour les IHM web



Alexandre.Demeure@inria.fr

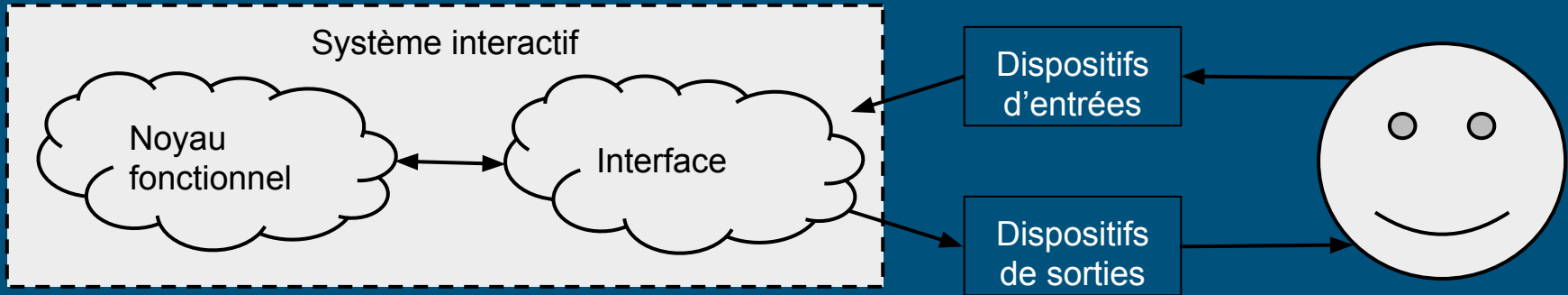


Pourquoi architecturer son code ?

- Assurer la modularité
- Avoir des blocs de codes aussi indépendants que possible
- Favoriser la réutilisation
- Favoriser la maintenance du code
 - Tests unitaires
 - Tests intégratifs

Architectures logicielles pour l'IHM

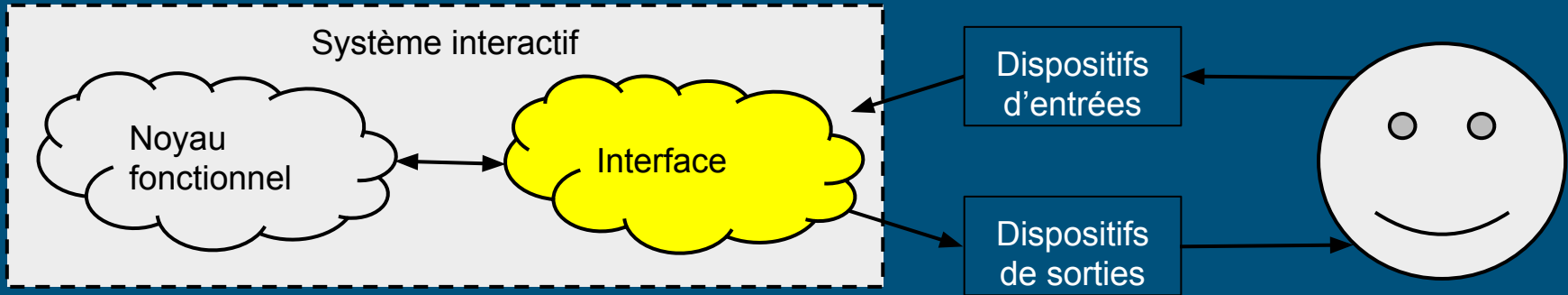
Un système interactifs comprend plusieurs éléments



Architectures logicielles pour l'IHM

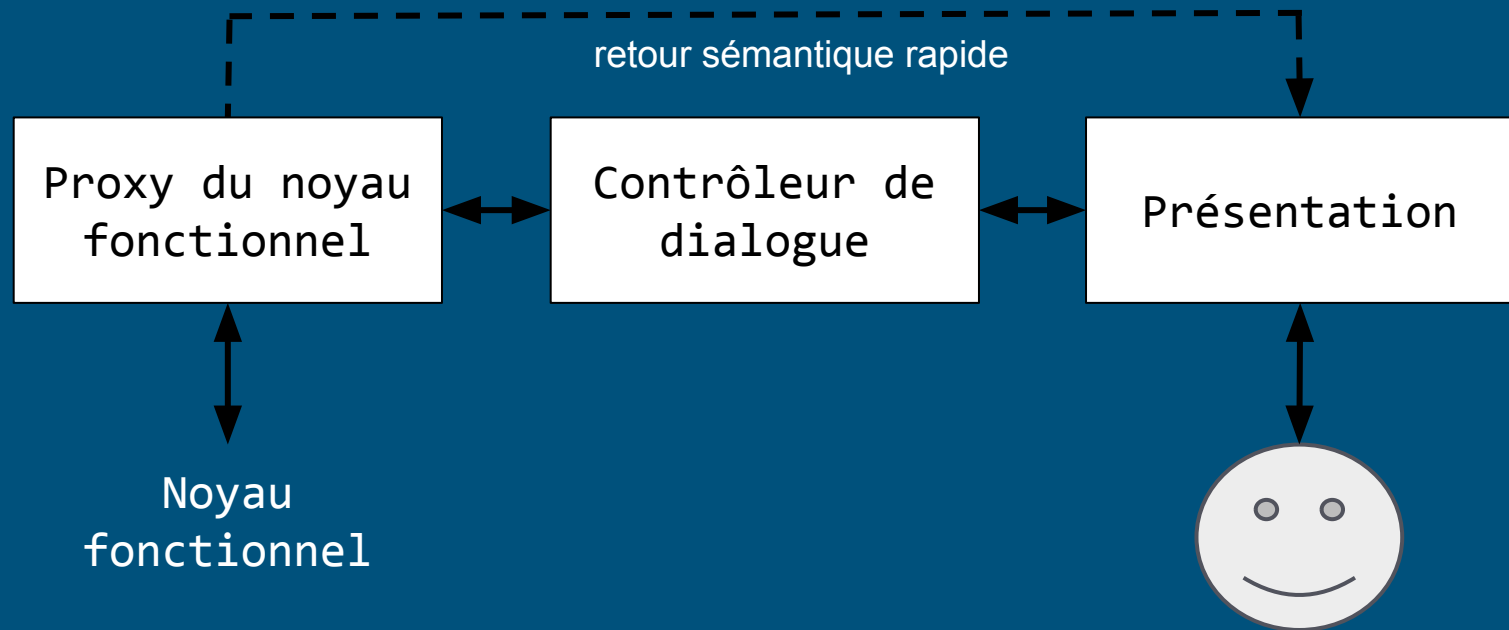
Un système interactifs comprend plusieurs éléments

Dans ce cours, nous nous concentrons sur le code de l'interface



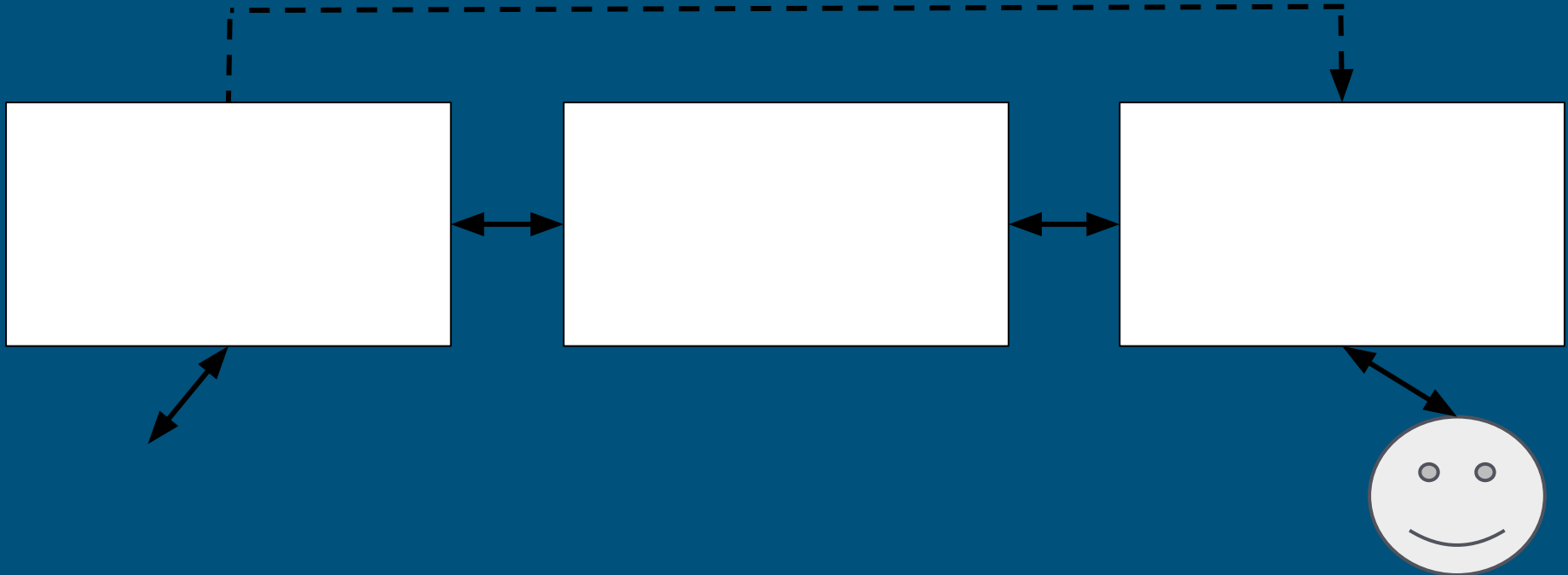
Modèle de Seeheim

Un point de vue linguistique sur l'interaction (sémantique, syntaxique, lexical)



Modèle de Seeheim

Illustration avec l'exemple de la liste de choses à faire :



Modèle de Seeheim

Illustration avec l'exemple de la liste de choses à faire :

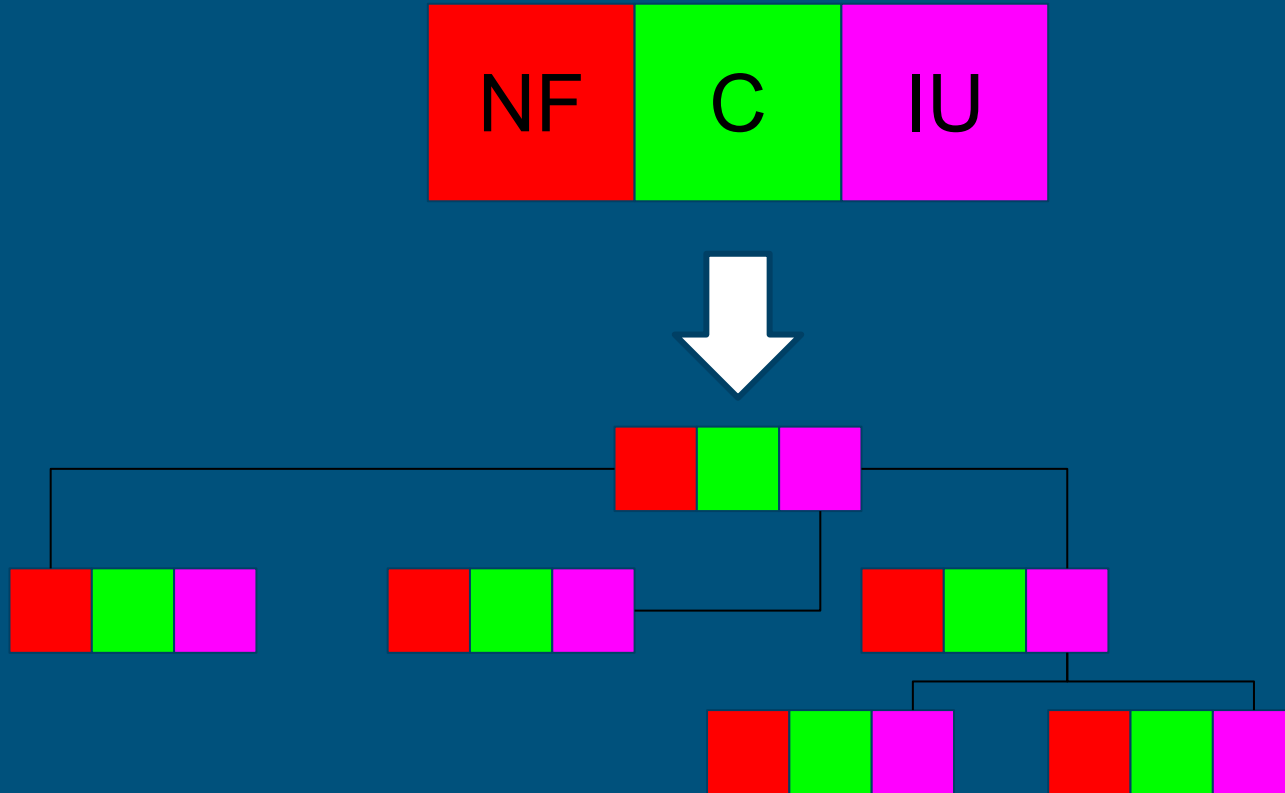


Modèle de Seeheim

Trop monolithique

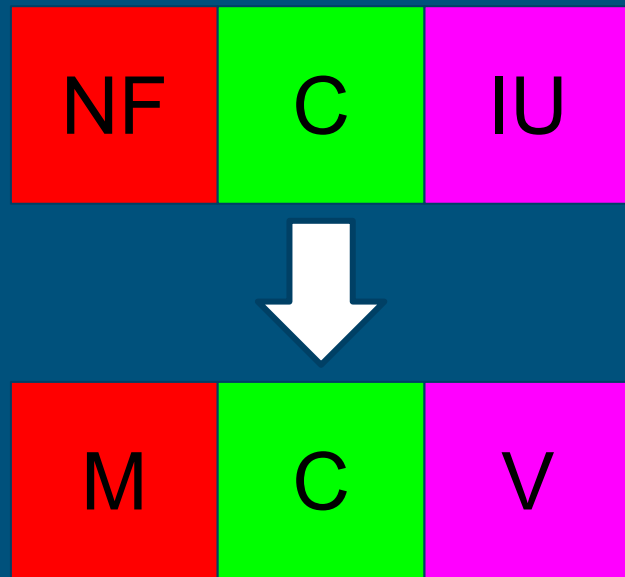
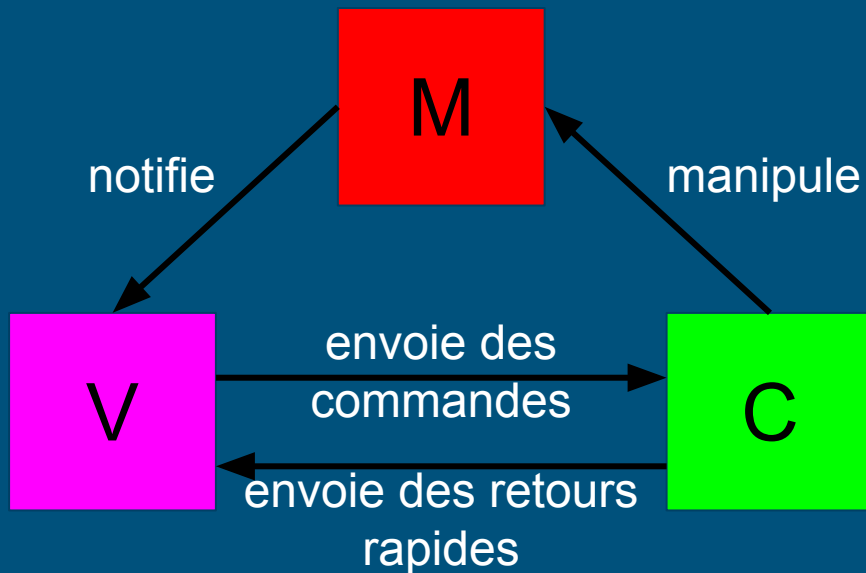
- Il faudrait pouvoir le décomposer dans le cas de gros projets
- Difficile à tester
- Difficile d'isoler des blocs réutilisables

Principe de décomposition



Modèle MVC

Model View Controller



Modèle MVC

Illustration avec l'exemple de la liste de choses à faire :

Modèle MVC

Illustration avec l'exemple de la liste de choses à faire : Chose à faire

Chose.js

Rendu HTML/CSS

Abonnement au modèle, traitement pour synchroniser l'affichage

Appel au contrôleur lors des actions utilisateurs

(ex: click sur supprimer)

Traduit les commandes de la vue en instructions pour le modèle

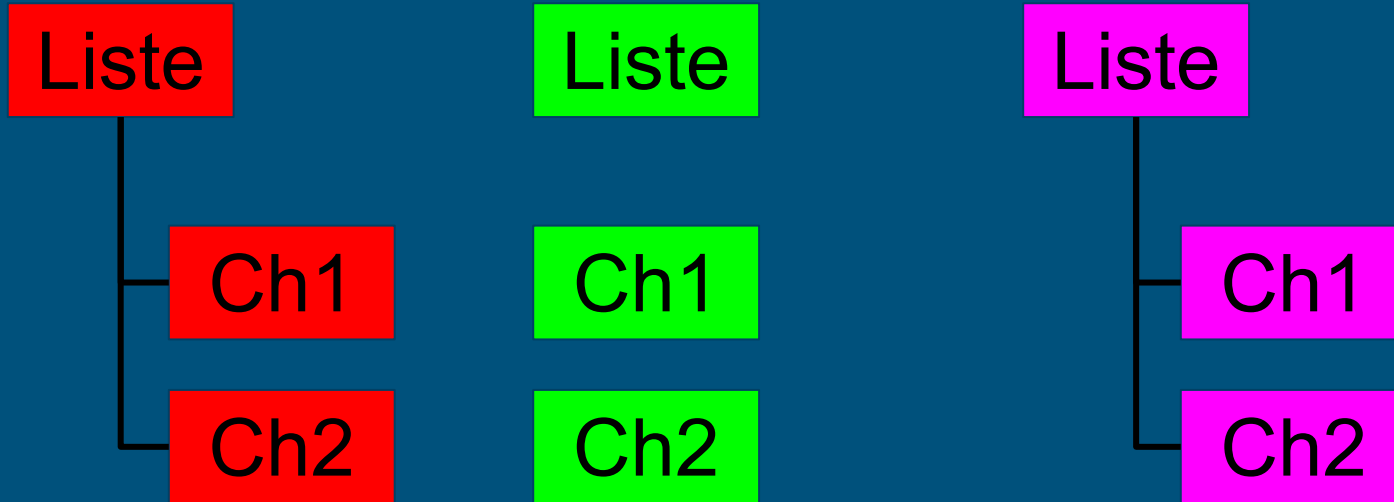
(ex: `obj.dispose()`)

Retours rapide d'information

(ex: traitement en cours)

Modèle MVC

Illustration avec l'exemple de la liste de choses à faire : Liste de Choses

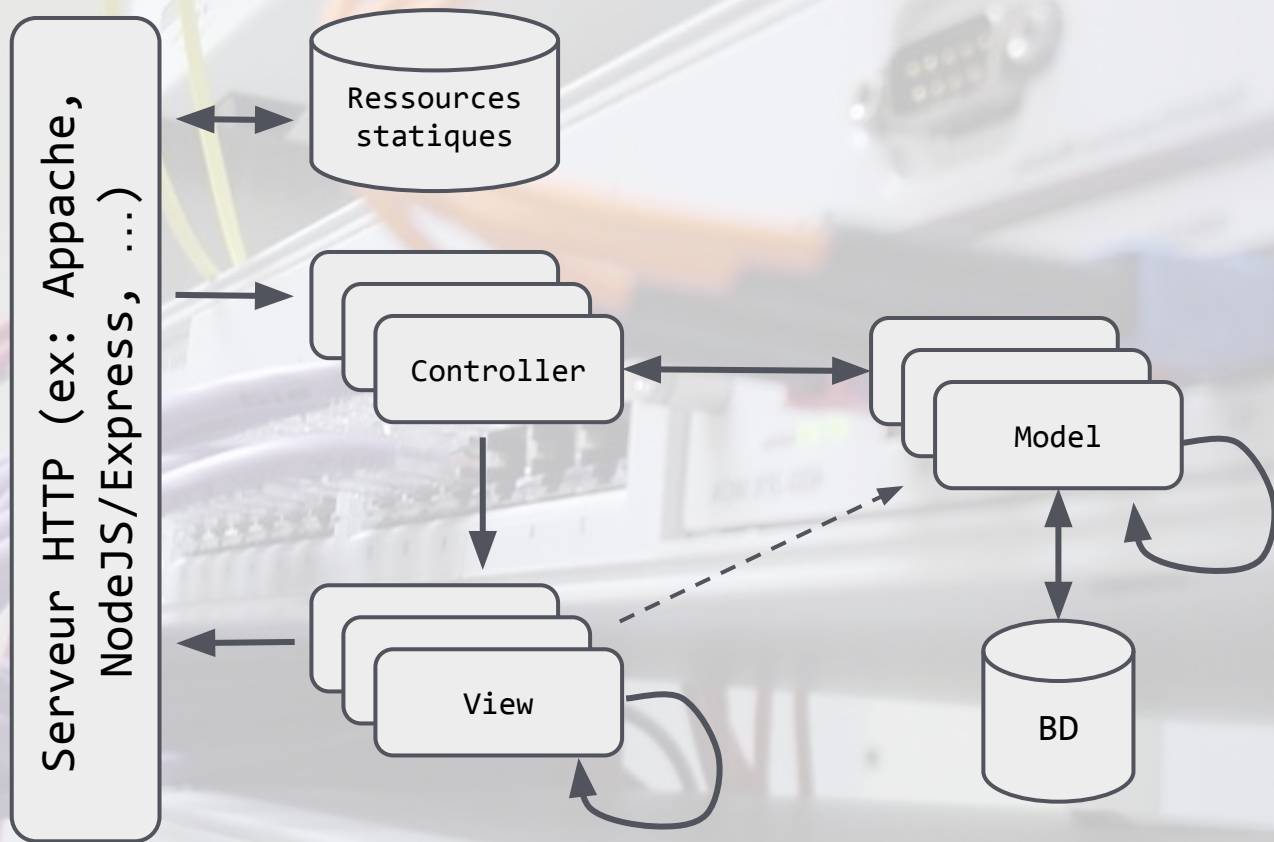


Modèle MVC

Requête
(HTTP, SOAP, ...)



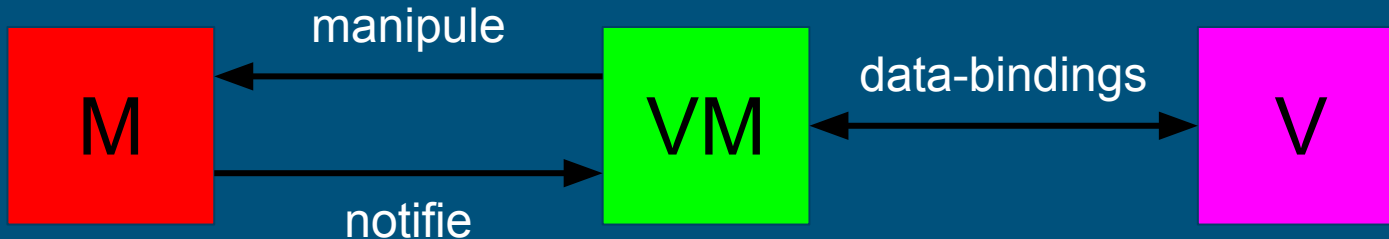
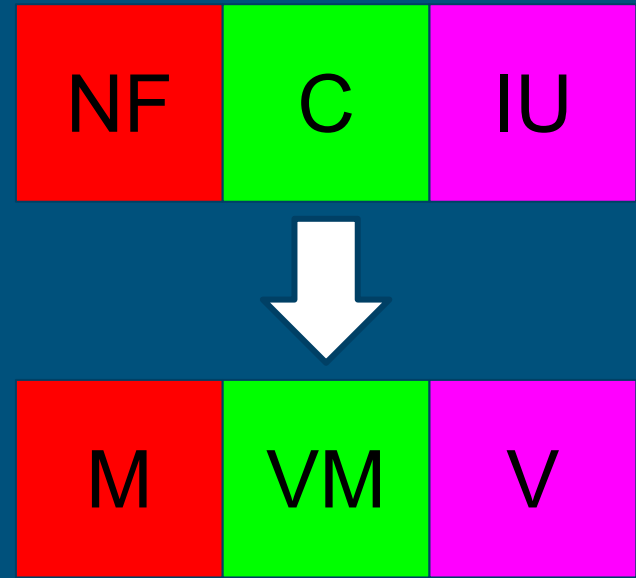
HTML, CSS, JS,
images, ...



Modèle MVVM

Originaire de Microsoft

- Vue en langage déclaratif (XAML, HTML/CSS)
- VM maintient
une structure de donnée intermédiaire
adaptée à la vue



Modèle MVVM

Illustration avec l'exemple de la liste de choses à faire :

Modèle MVVM

Illustration avec l'exemple de la liste de choses à faire :

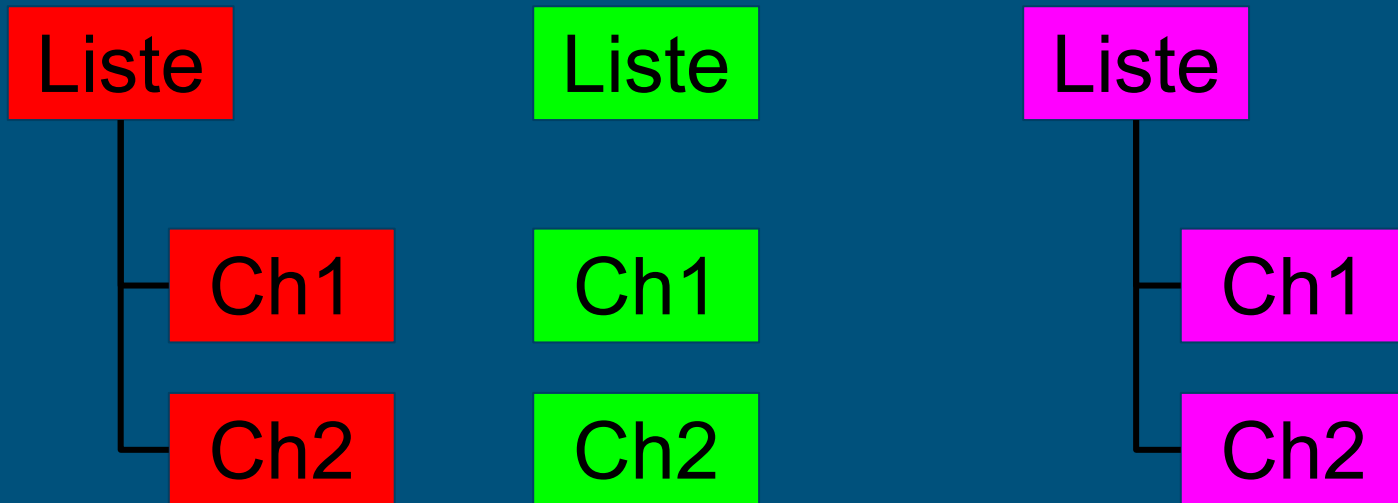
M

Synchronisation avec le M.
Maintient une structure de données dédiée à la vue.
Implémente une API pour la vue.

Description HTML de la vue au travers d'un template.
La vue est synchronisée avec la VM via le data-binding (automatique).
La vue appelle des commandes de la VM lors des interactions.

Modèle MVVM

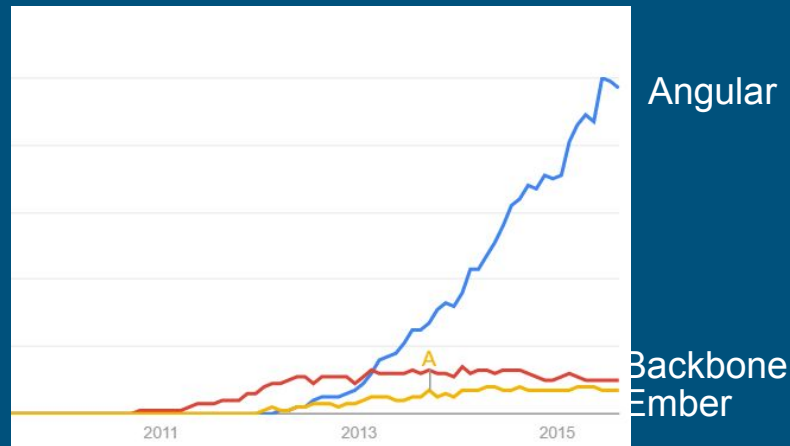
Illustration avec l'exemple de la liste de choses à faire : Liste de Choses



AngularJS

Framework développé chez Google

- Une communauté importante et active
- Ca évolue
 - version 1.5 : javascript, redéfinition
 - version 2 : en bêta, pensée pour Typescript mais utilisable en ES2015 et ES5
- Angular Material



Dans ce cours, on verra la version 1.5, en mettant l'accent sur les principes qui seront encore valides dans la version 2.

AngularJS 1.5

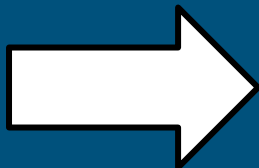
- Attention, si vous connaissez AngularJS, oubliez la variable `$scope`, ça disparaît (heureusement) avec la version 2
- Si vous ne connaissez pas AngularJS, vous n'avez rien entendu, `$scope` n'existe pas, c'est tabou...

AngularJS 1.5

Qu'aurait été HTML si il avait été pensé pour développer des applications ?

- Définir de nouvelles balises et de nouveaux attributs
- Utilisation du data-binding

```
<html>
...
<body>
<section class="listeChoses">
  <ul>
    <li><section class="Chose">...</li>
    <li><section class="Chose">...</li>
    <li><section class="Chose">...</li>
  ...
```



```
<html>
...
<body>
<liste-choses data="..."><liste-choses>
```

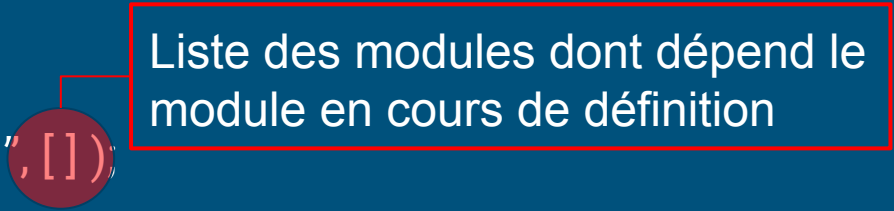
AngularJS 1.5

On définit des modules (≠ modules CommonJS)

Un module agrège des déclarations de composants et directives.

ex:

`angular.module("bshn", []);`



Liste des modules dont dépend le module en cours de définition

AngularJS 1.5

Les directives

- Définissent de nouvelles balises et de nouveaux attributs.
- Indiquent à angular comment compiler ces balises et attributs.
- Associent des objets javascript à ces balises et attributs.

```
var app = angular.module( "bshm", [ ] );
```

```
app.directive( "bshm-liste", function() {return {...
```

AngularJS 1.5

```
var app = angular.module( "bshm", [ ] );
```

```
app.directive( "bshm-liste", function() {return {
```

- restrict : 'E' pour balise ou 'A' pour attribut
- bindToController : true
- controllerAs : "ctrl", // Nom du controleur dans le template
- controller : function() { // Constructeur...// }
- template : La chaine de caractère du template HTML
- scope: un objet référençant les attributs qui seront utilisés

AngularJS 1.5

<bshm-liste

data="myDataObject"

titre="Ma liste de courses"

onmodified="console.log('hello')"

app.directive("bshm-liste", function() {return {

...

scope : { data: "<",
titre : "@",
onmodified : "&"
}

Le contenu de l'attribut data sera interprété comme un objet qu'il faudra synchroniser. Si il change, la vue change.

AngularJS 1.5

<bshm-liste

data="myDataObject"

titre="Ma liste de courses"

onmodified="console.log('hello')"

app.directive("bshm-liste", function() {return {

...

scope : { data: "<",

titre : "@",

onmodified : "&"

}

Le contenu de l'attribut titre sera interprété comme du texte.

AngularJS 1.5

<bshm-liste

data="myDataObject"

titre="Ma liste de courses"

onmodified="console.log('hello')"

app.directive("bshm-liste", function() {return {

...

scope : { data: "<",

titre : "@",

onmodified : "&"

}

Le contenu de l'attribut onmodified sera interprété comme un ensemble d'instructions. Ces instructions pourront être évaluées par la directive. Elle le seront dans le contexte où est placée la directive.

AngularJS 1.5

Les composants :

- Ce sont des directives spécialisées pour la définition de balises.
- Notion qui fait le pont avec Angular 2 où on ne parle plus que de composants

```
app.component( "bshmliste", {  
  template : ... //La chaîne de caractère du template HTML  
  bindings  : {nf: '<'}, // Equivalent à scope pour l'instruction directive  
  controller : function() { // Constructeur...// }
```

AngularJS 1.5

Revenons à l'exemple de la liste de choses à faire.