Retour d'expérience Django à Météo-France

Fabien MARTY <fabien.marty (at) gmail.com>
DjangoCon Toulouse, novembre 2012

METEO FRANCE
Toujours un temps d'avance

Introduction

Votre serviteur

- Fabien MARTY < fabien.marty (at) gmail.com >
  - Contributeur PHP/Zend à titre personnel...
    - ... pendant ma jeunesse
  - Architecte à Météo-France depuis 6 ans
  - Responsable technique d'un gros projet métier : **Synopsis**
  - Joue avec « Django » depuis 2 ans

Crédits : animuchan.net

Introduction

Météo-France (http://www.meteofrance.com)

- Un établissement public à caractère administratif
  - Forte composante technique et scientifique
  - 3500 personnes (dont 1200 à Toulouse)
- Sa mission principale :
  - Alerter les autorités et les populations des phénomènes météorologiques dangereux
- Une filiale MFI :
  - Commercialise nos solutions à l'étranger
  - Sponsorise cet événement !

METEO FRANCE
Toujours un temps d'avance

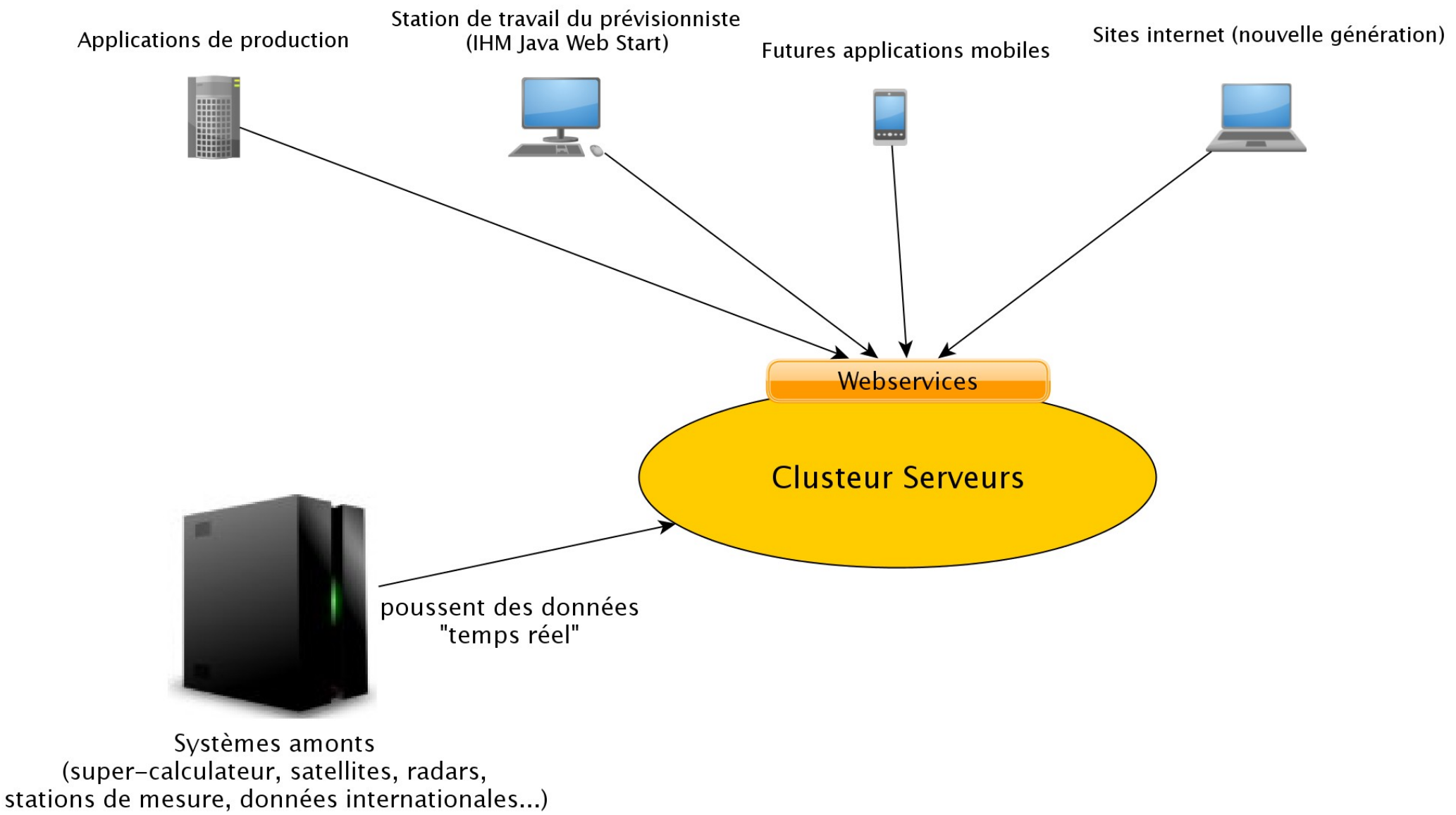
mfi
METEO FRANCE INTERNATIONAL

Le projet Synopsis

- Un gros projet métier :
  - Sur 5 ans
  - Charge estimée : 50 hommes.années (dev)
- Objectifs :
  - Nouvel outil unifié du prévisionniste opérationnel : *« Synergie Next »*
  - Architecture scalable et interopérable
- Cadre technique :
  - IHM « Java Web Start » multi-plateformes
  - Architecture orientée services sur clusteur Linux...

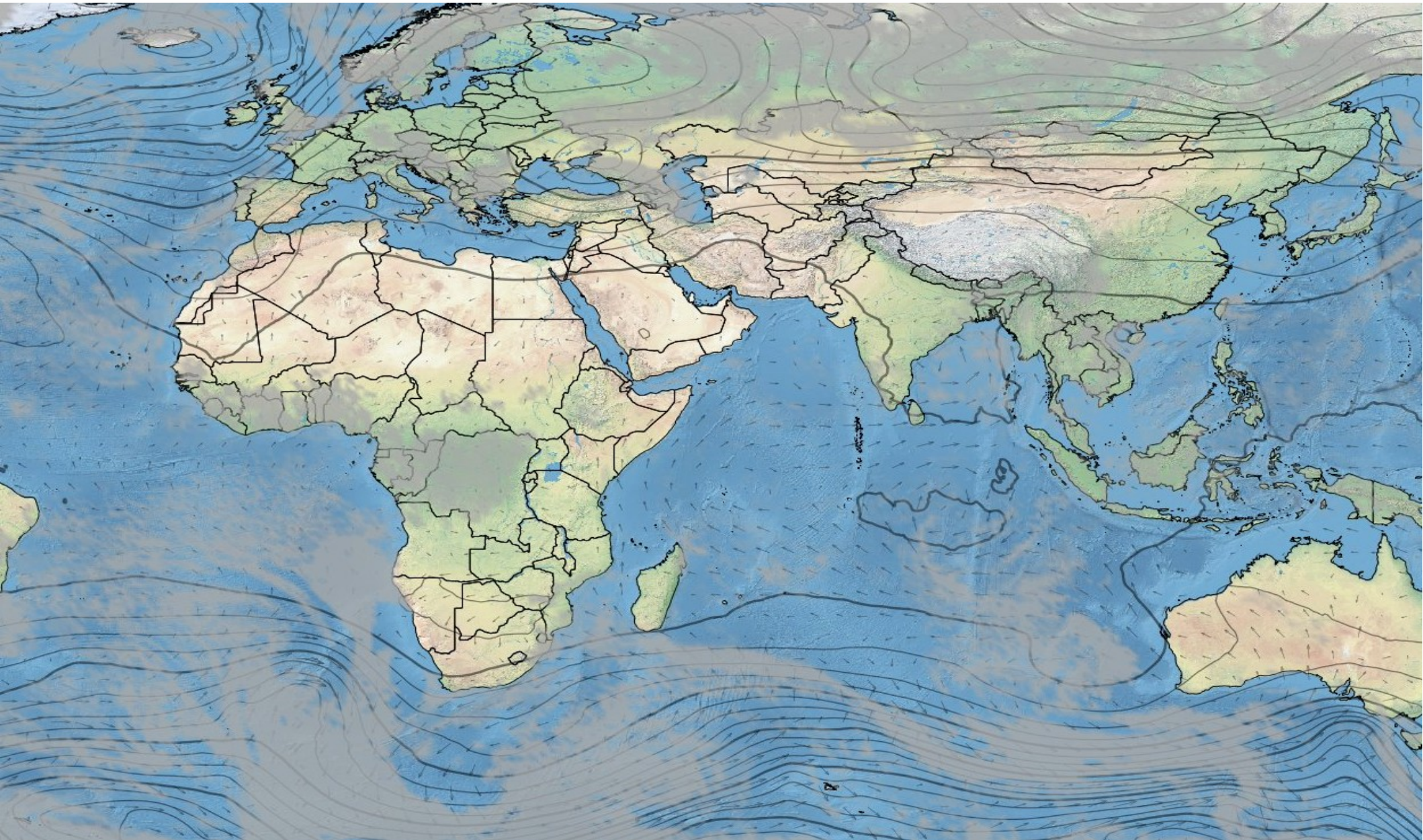
Introduction

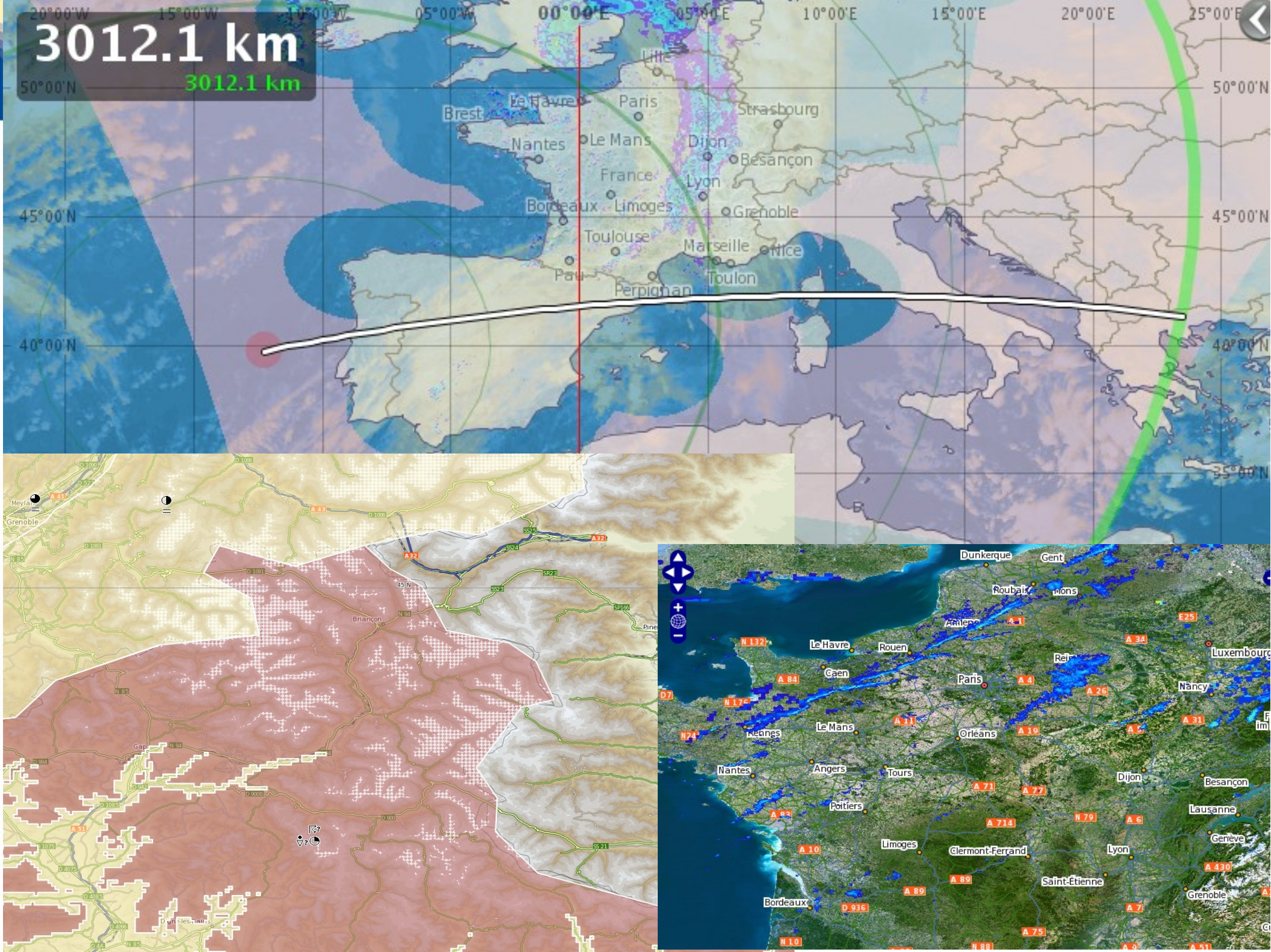
Le projet Synopsis

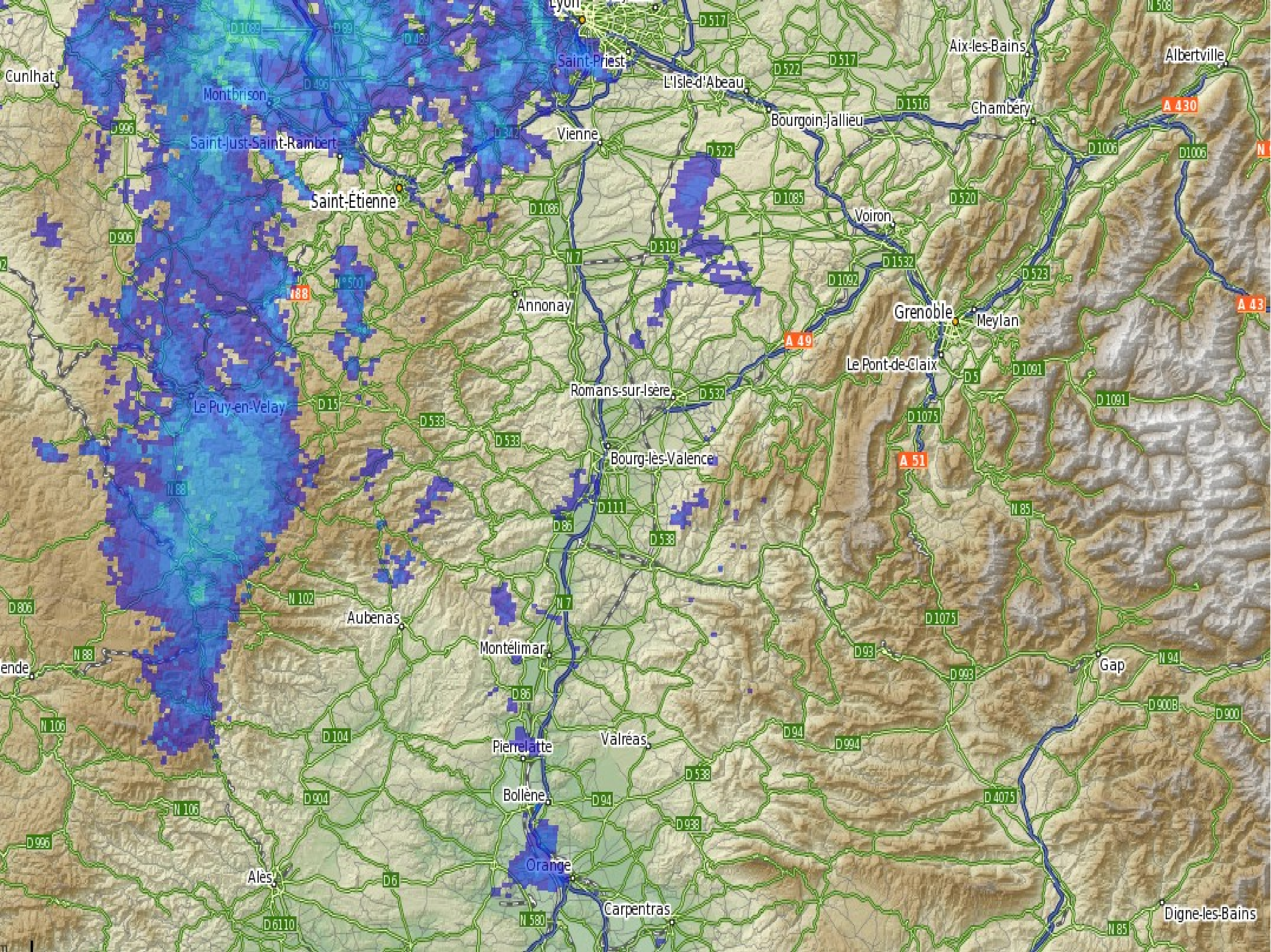


Introduction

Le projet Synopsis (exemples de tracés)




3012.1 km
3012.1 km
20°00'W
15°00'W
10°00'W
05°00'W
00°00'E
05°00'E
10°00'E
15°00'E
20°00'E
25°00'E
50°00'N
45°00'N
40°00'N
35°00'N
Lille
Le Havre
Paris
Brest
Strasbourg
Nantes
Le Mans
Dijon
Besançon
France
Lyon
Bordeaux
Limoges
Grenoble
Toulouse
Marseille
Nice
Pau
Toulon
Perpignan
Meylan
Grenoble
Briançon
Gap
Dunkerque
Gent
Roubaix
Mons
Amiens
Le Havre
Rouen
Caen
Paris
Reims
Luxembourg
Nancy
Le Mans
Orléans
Nantes
Angers
Tours
Dijon
Besançon
Poitiers
Lausanne
Genève
Limoges
Clermont-Ferrand
Lyon
Saint-Étienne
Grenoble
Bordeaux



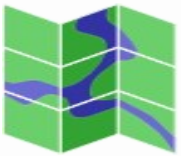
Lyon
D 517
N 508
Aix-les-Bains
Albertville
Cunlhat
D 1089
D 89
Saint-Priest
L'Isle-d'Abeau
D 522
D 517
Montbrison
D 496
Bourgoin-Jallieu
D 1516
Chambéry
A 430
D 996
Saint-Just-Saint-Rambert
Vienne
D 522
D 1006
D 1006
Saint-Étienne
D 1086
D 1085
D 520
Voiron
D 906
N 7
D 519
D 1532
D 523
N 88
D 1092
Annonay
Grenoble
Meylan
A 43
A 49
Le Pont-de-Claix
D 1091
Romans-sur-Isère
D 532
D 5
Le Puy-en-Velay
D 15
D 533
D 533
D 1075
D 1091
Bourg-lès-Valence
A 51
N 88
D 111
D 86
D 538
N 85
N 102
D 806
Aubenas
N 7
D 1075
Montélimar
D 93
Gap
N 94
N 88
D 86
D 993
D 900B
D 900
N 106
D 104
D 94
D 994
Pierrelatte
Valréas
D 538
D 904
Bollène
D 94
D 4075
N 106
D 938
D 996
Alès
D 6
Orange
Carpentras
D 6110
N 580
N 85
Digne-les-Bains

Le projet Synopsis

- Challenges techniques principaux
  - Scalabilité horizontale et performances interactives
    - « C10K problem »
    - 1 tracé = 1 seconde
  - Assimilation de données « temps réel »
    - Grande diversité (modèles numériques, satellites, radiosondages...)
    - Volume conséquent (de l'ordre de 500 Go par jour)
  - Services de tracés « à la demande »
    - Interopérable (normes OGC)
    - Gestion du cache très subtil
  - IHM Java
    - Navigation multi-dimensionnelles (4D et plus...)
    - Liberté de déplacement / zoom (« à la Google Maps »)

Le projet Synopsis

- Briques libres principales :
  - **PostgreSQL / PostGIS** + module DAV de **nginx** : stockage
  - **Redis** : bus orienté messages, cache, files d'attente, ...
  - **Nginx** : frontal web, cache de premier niveau
  - **Magics++** : outil d'isolignage/tracé météo
  - **MapServer** : solution de tracé de données géoréférencées
  - **GDAL** : le couteau suisse de l'imagerie géoréférencée



Le projet Synopsis

- Briques libres principales :
  - ... et **Django** (ouf !)

Historique

Historique

Au commencement il y a Python...

- La culture technique dominante (historique) à Météo-France, c'est :
  - « le C, le Shell et le Fortran »
- Au lancement du projet, on a cherché un langage :
  - De plus haut niveau
  - Avec des « bindings » de qualité pour les briques libres pré-senties
- **Python était le seul choix !**

Historique

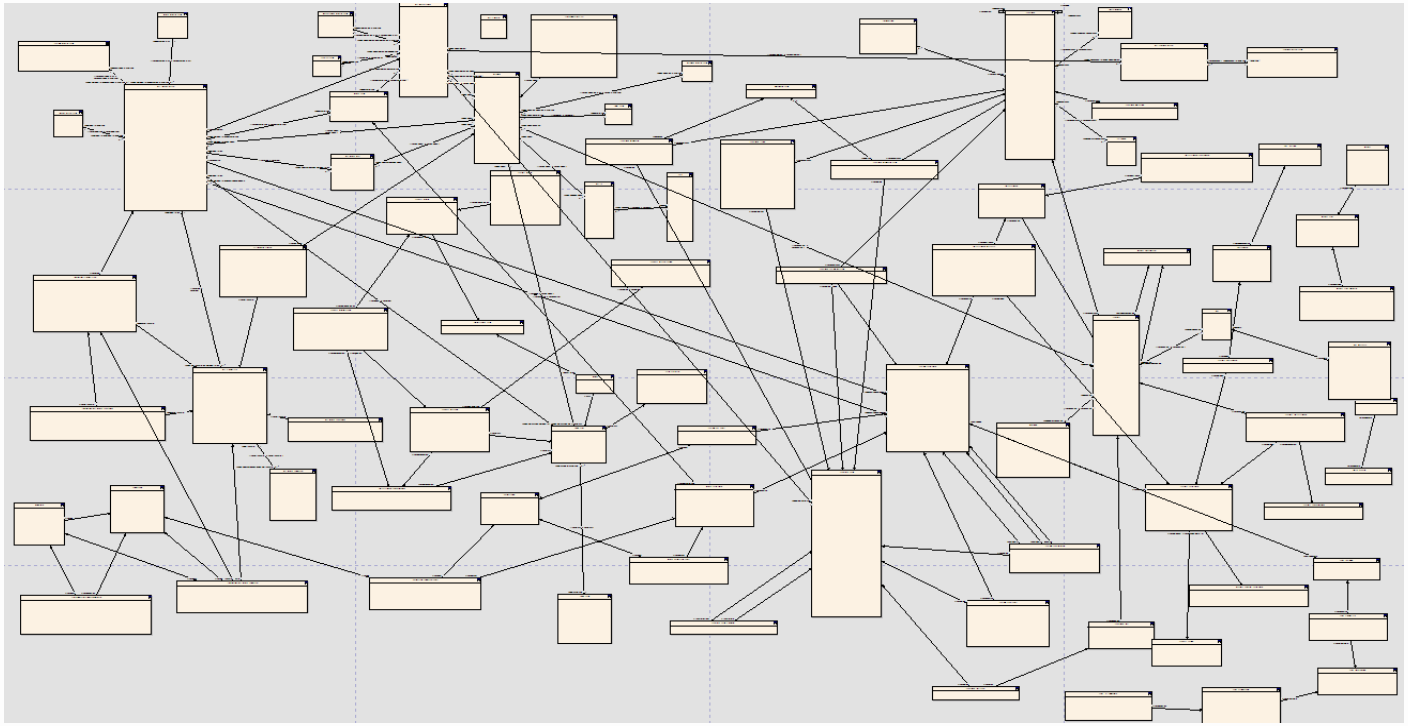
Puis le poney magique est arrivé au pays des grenouilles

- Tout est parti du modèle objet ISO-19115 (« geospatial metadata »)...
- Il nous fallait rapidement :
  - une interface d'admin
  - quelques webservices RESTful d'accès
- ...

Historique

Puis le poney magique est arrivé au pays des grenouilles

- ... mais le modèle UML ne rentrait pas dans une seule slide !

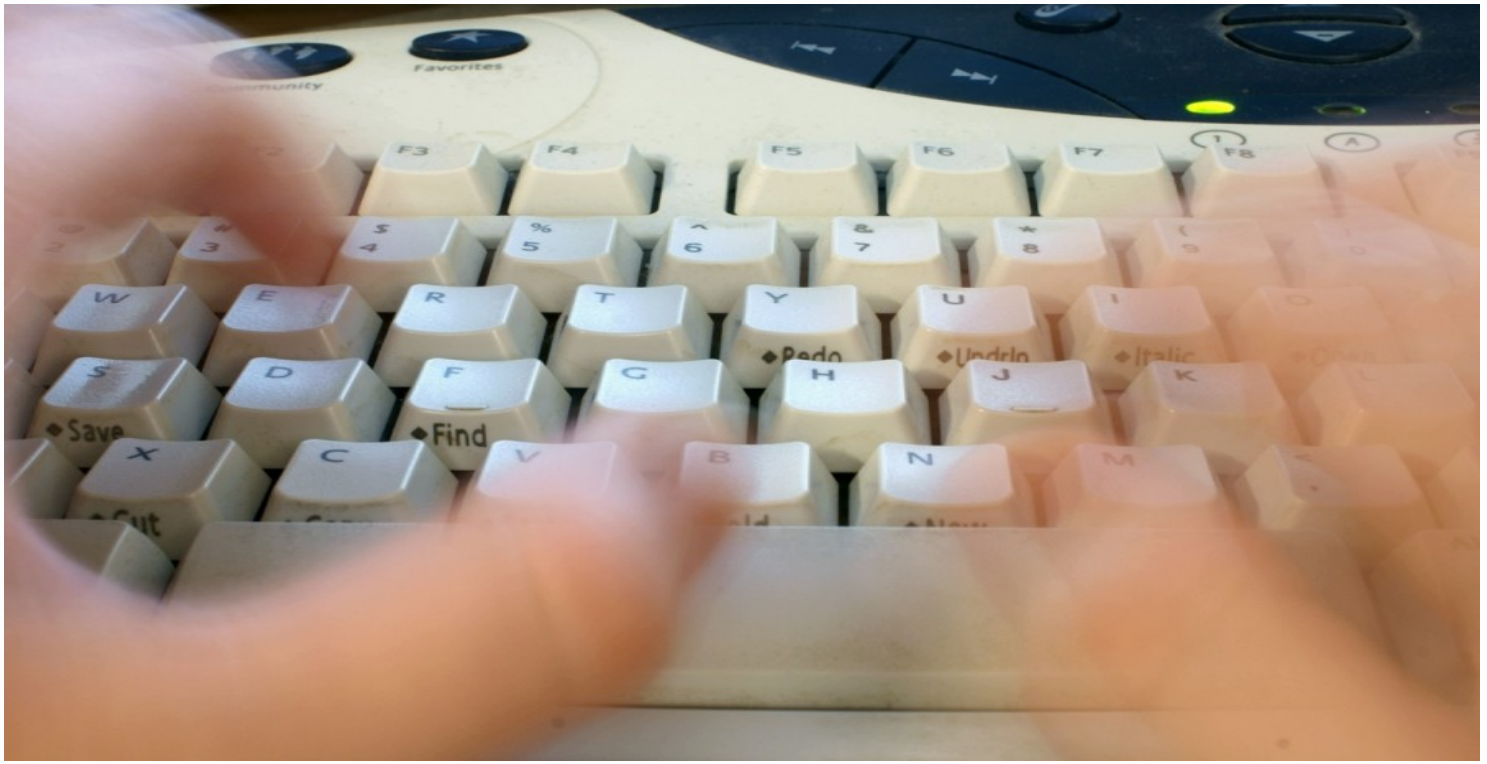


Puis le poney magique est arrivé au pays des grenouilles

- On a introduit Django sur ce seul périmètre :
  - Une trentaine de classes
  - Des liaisons dans tous les sens
- Au bout d'une semaine, on était déjà séduit :
  - Une documentation claire et complète
  - Pas de hacks nécessaires
  - Juste l'ORM et l'admin natif !
  - ... et on avait un prototype tout à fait fonctionnel

Puis le poney magique est arrivé au pays des grenouilles

- On a économisé des semaines de travail !



Crédit : Chris Metcalf

Puis le poney a grandit

- On a installé « grappelli »...
  - ...pour avoir un résultat un peu plus léché sur l'admin
- On a démarré une deuxième instance :
  - Minimaliste (pas d'ORM, pas de session...)
  - Pour servir de cadre à nos webservices métiers
  - Qu'on a progressivement interfacé avec notre :
    - Solution de log maison
    - Solution de cache maison
    - Architecture cluster maison
- **Et toujours pas « hacks » ni de « patchs custom » !**

Historique

Enfin le poney a finit par mettre les pieds sur la table

- L'architecture serveur est basée sur une dualité core/plugins
  - => nos plugins sont devenus des « class based views »...
  - ... qui hérite d'une classe « noyau »
- Un middleware custom est apparu...
  - ... pour gérer des subtilités de communication avec notre bus logiciel
- Toutes les pages web internes se sont subitement Django-isées
  - Portail web d'accès
  - Interface de monitoring
  - [...]

Historique

Enfin le poney a finit par mettre les pieds sur la table

- Le moteur de templates est utilisé à toutes les sauces...
  - ... pour les pages HTML bien évidemment
  - ... mais aussi :
    - pour fabriquer dynamiquement les « mapfiles » MapServer
    - **<chut>** pour fabriquer le fichier JNLP du client Java **</chut>**
    - **En dehors du contexte web !**

Historique

Enfin le poney a finit par mettre les pieds sur la table

```
cat ${HOME}/config/nginx.conf |env_sed.py >${HOME}/config_auto/nginx.conf
```

```
nginx -c ${HOME}/config_auto/nginx.conf
```

« Extrait adapté d'un shell de démarrage nginx sur le projet »

(utilisation Django en dehors du contexte web)

Historique

Enfin le poney a finit par mettre les pieds sur la table

Fichier de configuration nginx **avec tags Django**

Fichier de configuration « résolu »

```
cat ${HOME}/config/nginx.conf |env_sed.py >${HOME}/config_auto/nginx.conf
```

Wrapper Django qui initie un contexte de template à partir de l'environnement

```
nginx -c ${HOME}/config_auto/nginx.conf
```

On lance « nginx » en utilisant le fichier de configuration résolu

« Extrait adapté d'un shell de démarrage nginx sur le projet »

(utilisation Django en dehors du contexte web)

Historique

Enfin le poney a finit par mettre les pieds sur la table

- En cours :
  - Introduction de « GeoDjango »
  - Utilisation de « haystack » pour :
    - La recherche fulltext
    - Certains mécanismes d'auto-complétion

Bilan

Bilan

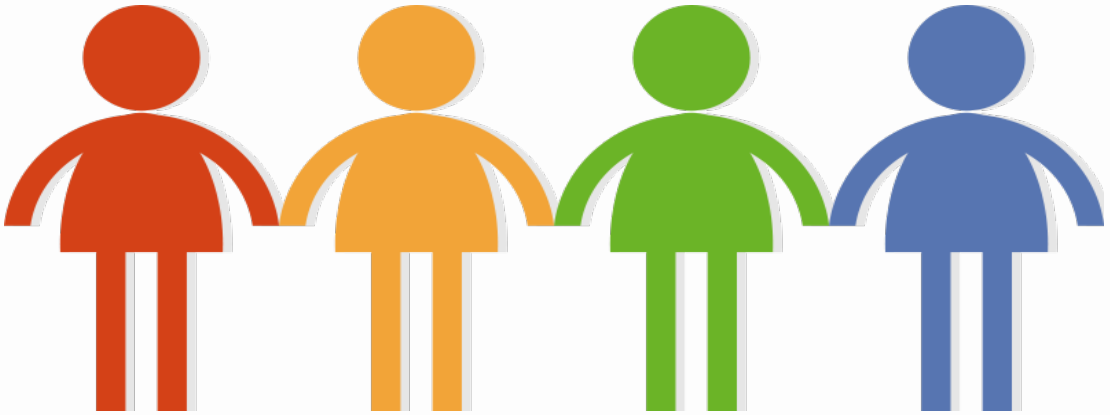
Les forces de Django

- La documentation claire et bien fournie...
  - ... il y a même de vrais livres ;-)
- La couche ORM et son intégration dans l'admin
- **=> Un ticket d'entrée assez faible**

Bilan

Les forces de Django

- La flexibilité
- La possibilité de faire les choses proprement sans hacks
- La stabilité des APIs
- **L'écosystème et la communauté !**



Bilan

Les faiblesses de Django

- Les fixtures et/ou la sérialisation
  - L'enfer de la « pk autoincrement »
  - La difficulté de « dumper » un objet et toutes ses dépendances
  - La lenteur du mécanisme
  - Le manque d'outillage en natif...
    - ... même si « django-fixture-magic » aide un peu

Bilan

Les faiblesses de Django (et nos contournements)

- Les fixtures et/ou la sérialisation
  - L'enfer de la « pk autoincrement »
    - **=> Remplacement par une pk de type « uuid »**
  - La difficulté de « dumper » un objet et toutes ses dépendances
    - **=> « django-fixture-magic » est un bon début**
    - **=> + hacks spécifiques**
  - La lenteur du mécanisme
    - **=> pas encore creusé**
  - Le manque d'outillage en natif...
    - **=> « django-fixture-magic » est un bon début**
    - **=> quelques compléments maison en cours de chantier**

Conclusion

METEO FRANCE
Toujours un temps d'avance

Conclusion

- **On est globalement très satisfait...**
- ... même s'il reste quelques points mineur à traiter

- De notre point de vue, Django est à la fois :
  - Rigide => ce qui donne un cadre
  - Flexible => car on peut prendre en compte des spécificités sans hacks et s'adapter aux changements

- **Une dualité finalement assez renversante !**

Crédit : Ibrahim Lujaz

Fin

Questions ?

METEO FRANCE
Toujours un temps d'avance