RAINCODE

L’obsolescence des systèmes informatiques

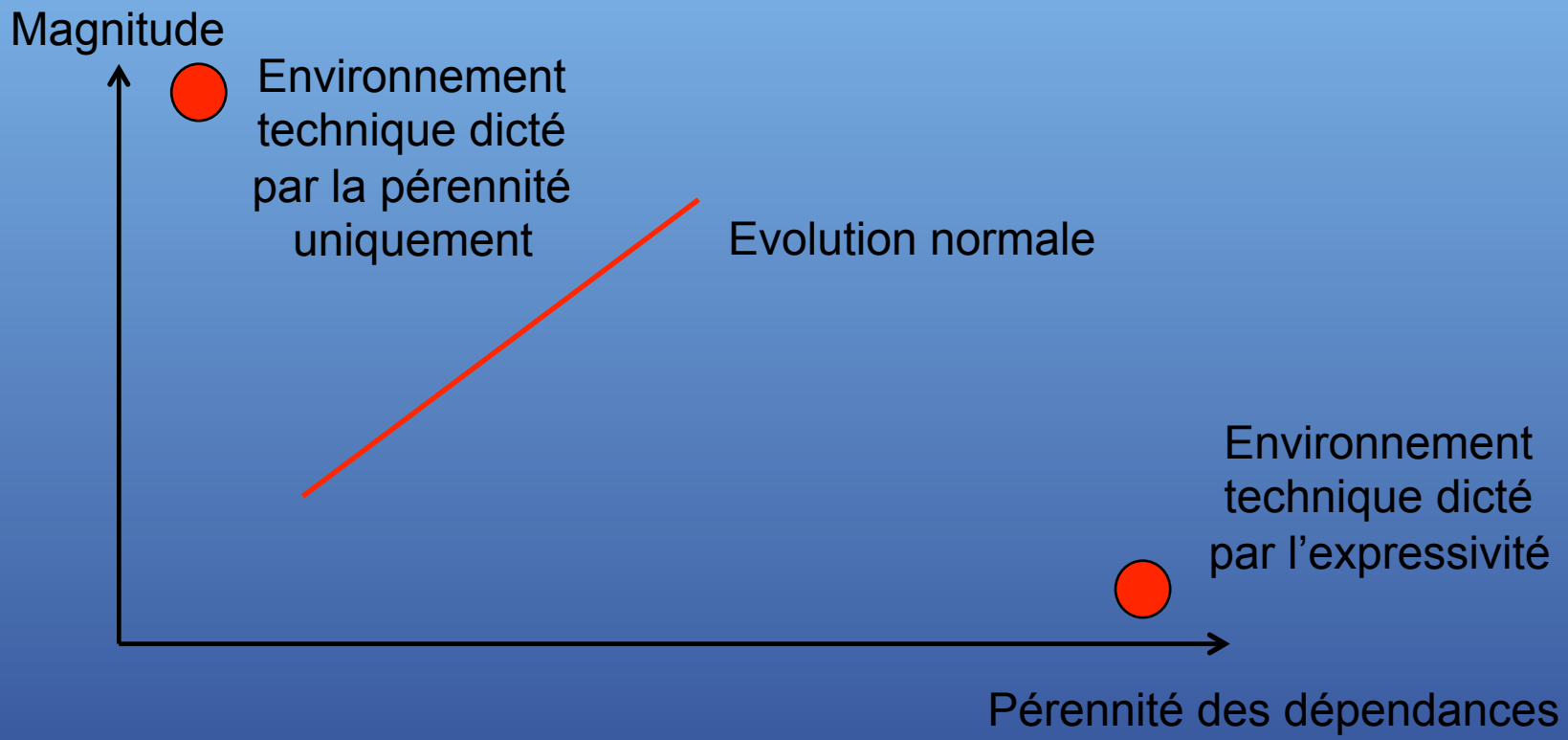
RAINCODE

RainCode, en 1 slide

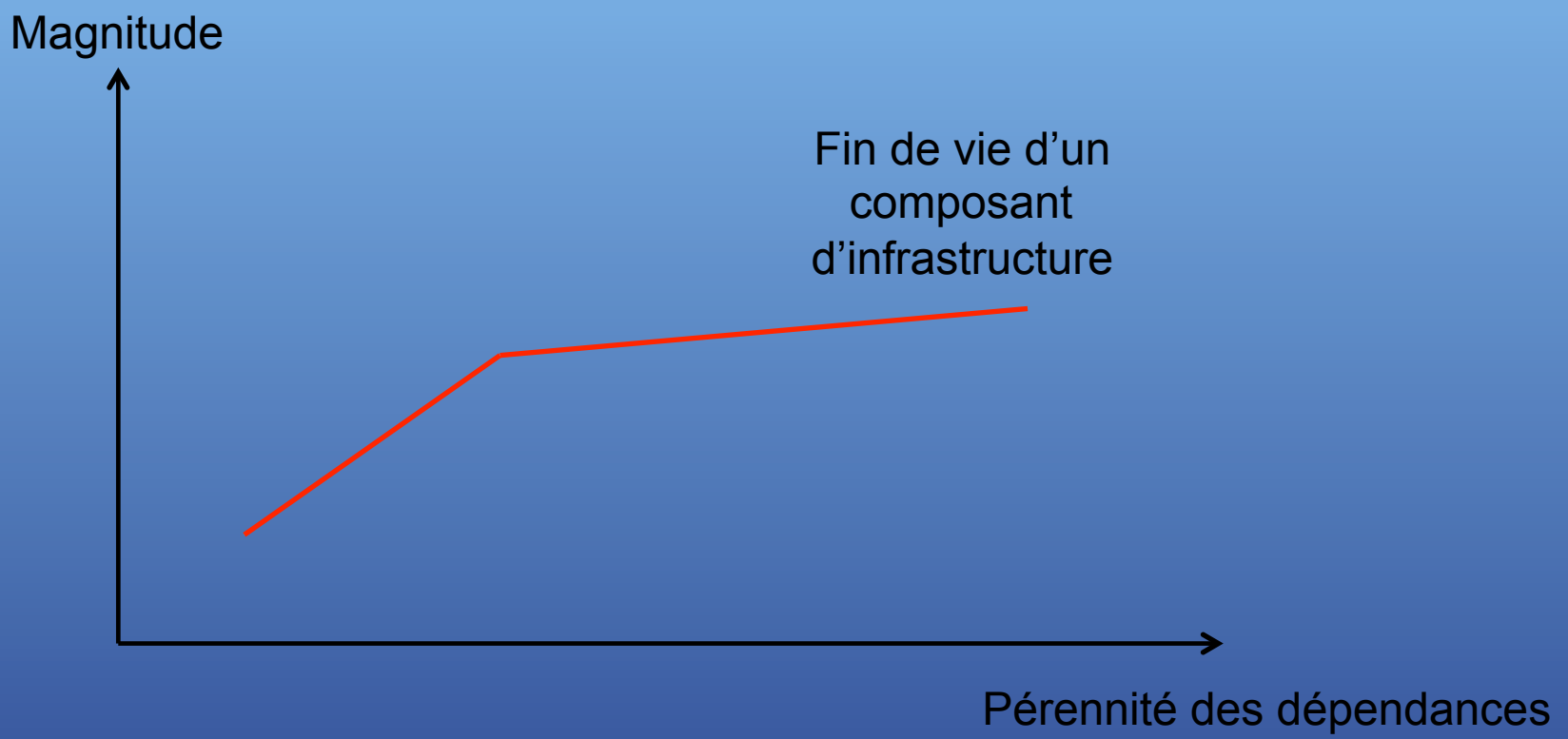
- Basée à Bruxelles
  - Fondée en 1998
- 20 personnes
  - Une majorité d'ULBistes
  - Plus de 2/3 technique
- Spécialiste des langages de programmation
  - Compilation
  - Migration
  - Analyse
  - Etc.

- Internationalisation
  - Structure de vente en France
  - Filiale aux Etats-Unis
- Clients dans le monde entier
  - Japon, Etats-Unis, Danemark, Canada, Pays-Bas, France, Italie, etc.
- Partenariats
  - Microsoft, Oracle, IBM …
- (Et nous recrutons activement)

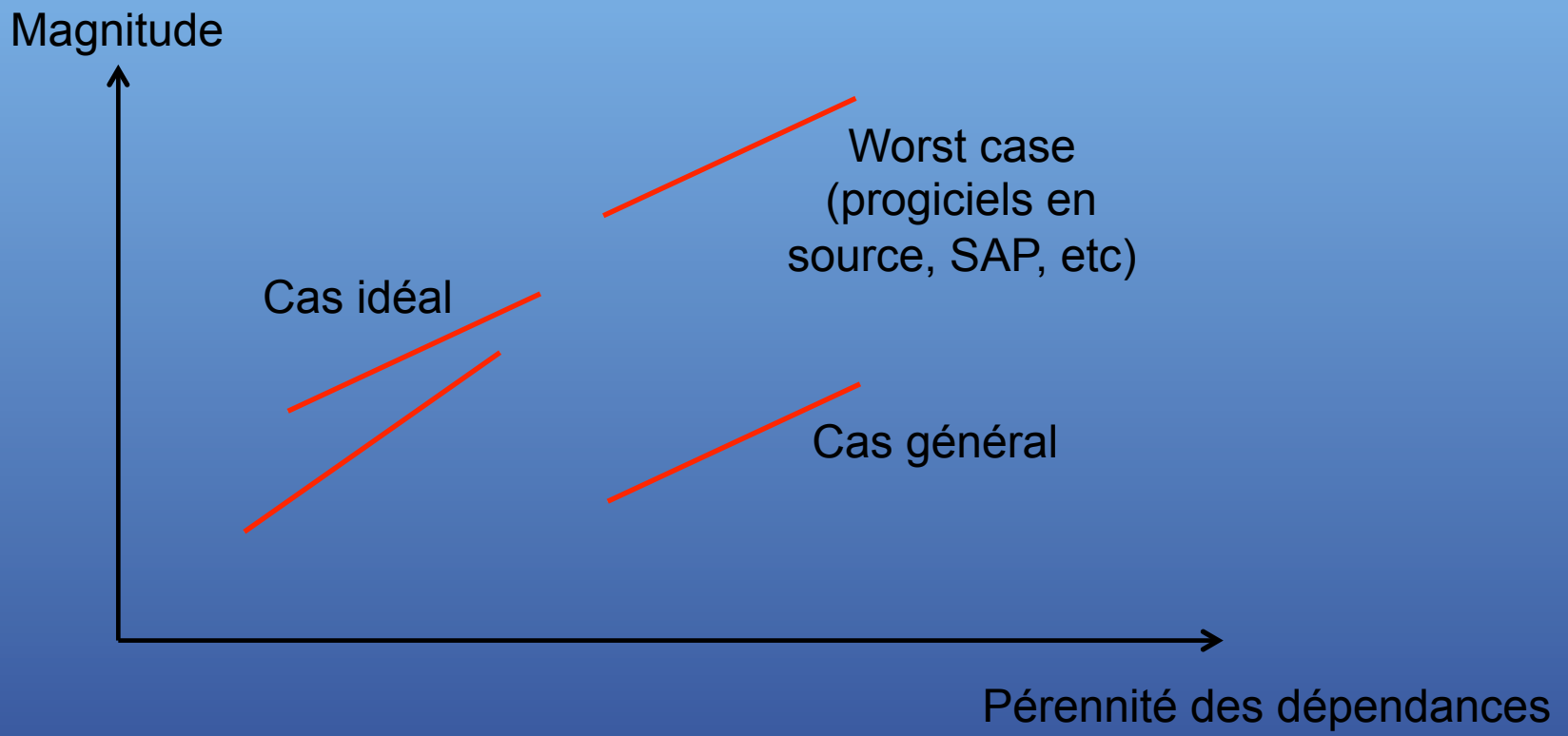
Le coût d'un système sur 2 axes

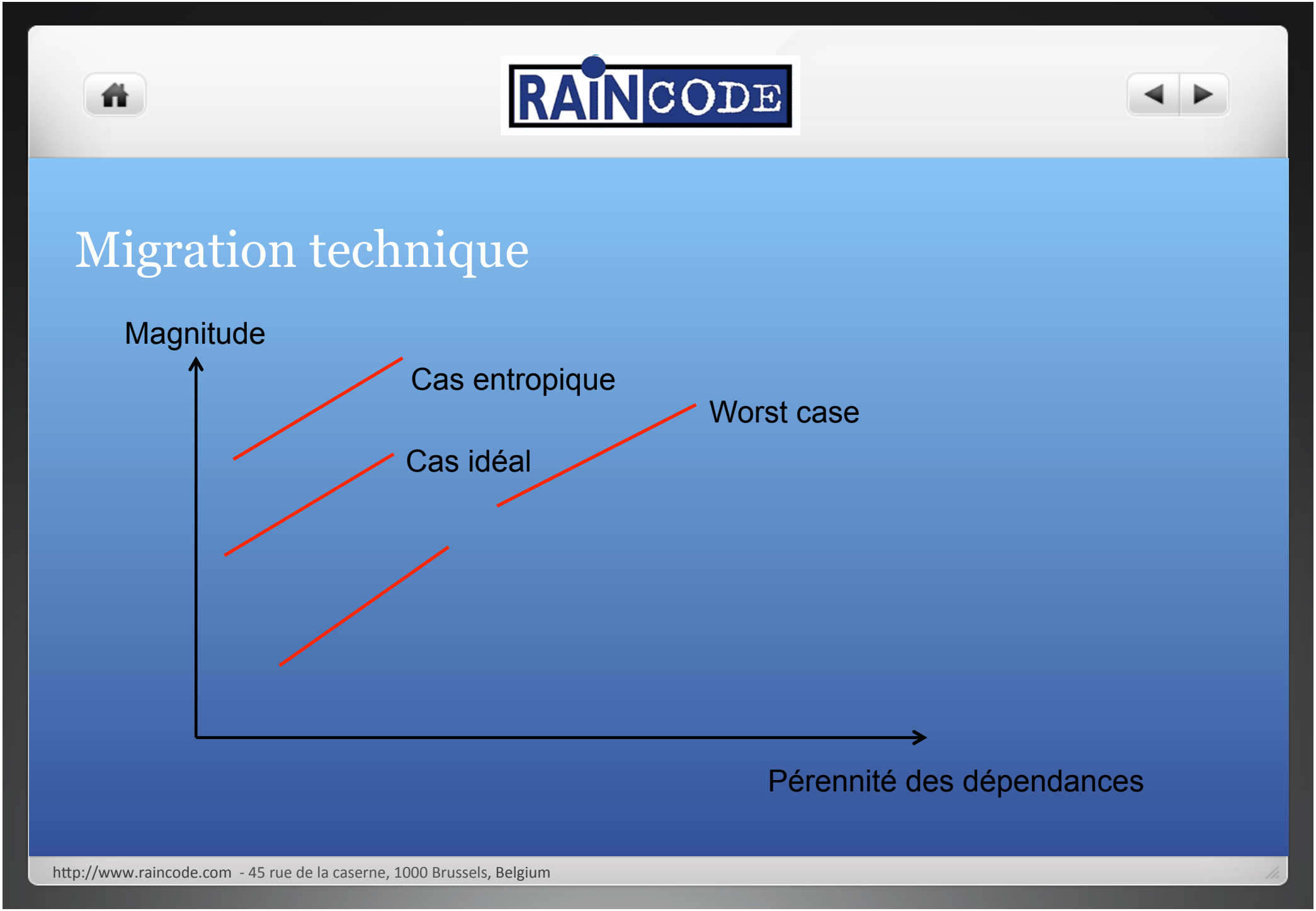


Evolutions plus chaotiques



Installation d'un progiciel




RAINCODE
Migration technique
Magnitude
Cas entropique
Worst case
Cas idéal
Pérennité des dépendances
http://www.raincode.com - 45 rue de la caserne, 1000 Brussels, Belgium

RAINCODE

Quid du redéveloppement?

- Pourquoi tant de frilosité?
  - Yaka redévelopper les applications tous les 10-15 ans
  - S'affranchir des contraintes temporelles
  - Intuitivement raisonnable
- Facteur 1: le volume
  - Ces applications on été développées sur 20 ans, la masse fonctionelle est importante
  - Redévelopper prend 3 à 5 ans, et même cela est optimiste
- Facteur 2: le prix
  - Largement suboptimal
- Facteur 3: continuité
  - L'application doit continuer à évoluer pendant le redéveloppement
  - Cible mouvante
  - Données,
  - Intégration dans les systèmes extérieurs
- Au bilan…
  - Les projets de redéveloppement semblent échouer encore plus souvent que les projets de simple développement

Mort naturelle?

- Avant de s'inquiéter de l'obsolescence…
  - De quel horizon parlons-nous ?
- Combien de temps une application vit-elle?
  - Avant de tomber en désuétude ?
  - Avant que son utilité disparaisse?
  - Ou devienne marginale ?

- Selon le domaine ?
  - Industriel?
  - Bancaire ?
  - Assurance ?
- Pas le recul nécessaire…
  - Pour les applications bancaires ou d'assurances, cette durée de vie pourrait bien faire 100 ou 150 ans
  - Mais il faut attendre pour le savoir
  - Au moins 100 ou 150 ans ☺

La croissance pour tout paradigme

- Tojours plus, juste parce que le matériel le permet
- Toute la formation est dirigée vers le développement de nouveaux systèmes
  - Un tout petit peu sur la maintenance
  - Et rien du tout sur la fin de vie
- Un biais qui n'est pas propre à l'informatique
  - Le marketing est un autre exemple: introduction de nouveaux produits par rapport à la maintenance d'un marché existant

- Biais compréhensible
  - Devoir vivre avec les douleurs et erreurs du passé
  - Plutôt que de voir le futur en rose
  - Cécité sélective

RAINCODE

Croissance perpétuelle

- Des organisations qui croulent sous le poids de leur informatique
- Qui en meurent parfois
  - Même si ce n'est pas la version officielle
  - Winterthur
- Des systèmes trop complexes
- Groupe SNCB:
  - 150 millions d'euros pour l'implémentation d'un SAP HR, 40.000 employés
  - 4.000 euros par employé
  - Pour le privilège de pouvoir être payé

- La course aux armements applicative a déja eu lieu
  - Opérateurs téléphoniques
  - Banques pour particulier
- Mais le pire est à venir
  - Administration publique
  - La probabilité d'exactitude avoisine zéro
  - Facturation INAMI
- Et certains succès formidables
  - Les Miles des compagnies d'aviation

Limiter la croissance ?

- Offshore
  - La rationalité coût bénéfice n'est pas avérée
  - Moins une menace pour l'ecosystème que l'expatriation des centres de décision
  - Comme essayer de résoudre un problème NP-complet sur une machine plus puissante
- L'utilisation de progiciels
  - Externalisation de la complexité
  - Avec les limites vues plus haut
- Rationalisation
  - Bonne chance !

- Accepter la baisse de service
  - Détail de la facture Proximus
  - Tout a un coût, plus n'est pas toujours mieux…
- L'abstraction est un vecteur de réduction
  - Mais il augmente (généralement) le passif technique
  - Car malheureusement, abstraction et pérennité ont corrélés négativement

Dépendances applicatives

- Problème mal maitrisé
  - Parce que vendu par des vendeurs aux quotas trimestriels
  - Et géré pour les 20 ans qui viennent par ceux qui les écoutent
- La tectonique est plus violente que l'on ne pense
  - Personne n'est insubmersible
  - Ni SAP, ni IBM, ni Oracle, ni Google
  - Sur une échelle de 20 ans, il n'existe pas de garantie absolue (Compaq, Informix, Digital, Lotus)
- Ne pas croire que cela soit limité aux grosses applications mainframe
  - Et les applications SaaS ne font qu'accelérer le processus (Google wave)
  - Et ce n'est que le début
  - Twitter, Facebook, MySpace, Salesforce, Gmail, constituent des passifs incontrôlés

Dépendances techniques

- Les abscisses de nos graphes
- Couvre différents aspects
  - Langage
  - Format de données
  - Système d'exploitation
  - Moniteur transactionel
  - (Librairie particulière, comme ISPF)
- C'est le coeur de métier de RainCode

- Des sujets pas populaires
  - Désaffection des élites du technique vers le "business"
  - Mais ultimement, les systèmes qui tournent ne sont que des bits et des bytes
  - Sujets importants, complexes, et des enjeux économiques monstrueux

Quiz...

Si vous deviez développer un système devant vivre 30 ans, quel langage(s) utiliseriez-vous?

Cas 1: Java

- Est ce que Java
  20 ans?
  - Sans doute
- Est ce qu'une a
  pouvoir aiséme
  - Probablement pa

```
public void f90Ao00W_M640 () {
wAo00.fill(" ");
wAo00.setValue(aoFile.getAo00());
udFile.setUd00Undkey(aoFile.getAo00Undkey());
udFile.setUd00Recnum(1);
if ( ! (aoFile.getAo00Rechdr() > 450)) {
udFile.setUd00Rechdr(aoFile.getAo00Rechdr());
} else {
udFile.setUd00Rechdr(450);
}
udFile.insert();
udFile.setUd00Rechdr(udFile.getUd00Rechdr() + 15);
udFile.setUd00Half(wAo00.getWAo00Half1());
wssBegin.setIk("0");
BaseService fbBS;
fbBS = ServicesFactory.create("com.whoever.fb", this);
fbBS.addCallParameter(17, false);
fbBS.addCallParameter(fbActions.getFieldfbProgram(),
false);
fbBS.addCallParameter(fbActions.getFieldfbWrite(),
false);
fbBS.addCallParameter(fbActions.getFieldNullInfosStatus(),
false);
fbBS.addCallParameter(fbWritePositions4, false);
fbBS.execute();
if (statusArea.getFieldOneUd00Status().
isGreaterThan("02") &&
!statusArea.getFieldOneUd00Status().equals("22") &&
!statusArea.getFieldOneUd00Status().equals("23")) {
w_perform("secud_M720");
}
if (statusArea.getFieldOneUd00Status().equals("22") ||
statusArea.getFieldOneUd00Status().equals("23")) {
wssBegin.setIk("1");
}
udFile.insert();
if (wssBegin.getFieldIk().isFilled('0') &&
aoFile.getAo00Rechdr() > 450) {
udFile.setUd00Undkey(aoFile.getAo00Undkey());
```

ts:

), Jboss,

ncertées

oué

érie de

posants à

me cible

posée

ra inopérante

Cas

- V
  s
- D
  V
- O
- U
  -
  -
  -
  - Impacte potentiellement 25% de la population des Etats-Unis

```
%DTC
%DTC ; SF/XAK - DATE/TIME OPERATIONS ;1/16/92  11:36 AM
     ;;19.0;VA FileMan;;Jul 14, 1992
     D    I 'X1!'X2 S X="" Q
     S X=X1 D H S X1=%H,X=X2,X2=%Y+1 D H S X=X1-%H,%Y=%Y+1&X2
     K %H,X1,X2 Q
     ;
C    S X=X1 Q:'X  D H S %H=%H+X2 D YMD S:$P(X1,".",2) X=X_"."_$P(X1,".",2) K X1,X2 Q
S    S %=%#60/100+(%#3600\60)/100+(%\3600)/100 Q
     ;
H    I X<1410000 S %H=0,%Y=-1 Q
     S %Y=$E(X,1,3),%M=$E(X,4,5),%D=$E(X,6,7)
     S %T=$E(X_0,9,10)*60+$E(X_"000",11,12)*60+$E(X_"00000",13,14)
TOH  S %H=%M>2&'(%Y#4)+$P("^31^59^90^120^151^181^212^243^273^304^334","^",%M)+%D
     S %='%M!'%D,%Y=%Y-141,%H=%H+(%Y*365)+(%Y\4)-(%Y>59)+%,%Y=$S(%:-1,1:%H+4#7)
     K %M,%D,% Q
     ;
DOW  D H S Y=%Y K %H,%Y Q
DW   D H S Y=%Y,X=$P("SUN^MON^TUES^WEDNES^THURS^FRI^SATUR","^",Y+1)_"DAY"
     S:Y<0 X="" Q
7    S %=%H>21608+%H-.1,%Y=%\365.25+141,%=%#365.25\1
     S %D=%+306#(%Y#4=0+365)#153#61#31+1,%M=%-%D\29+1
     S X=%Y_"00"+%M_"00"+%D Q
     ;
YX   D YMD S Y=X_% G DD^%DT
YMD  D 7 S %=$P(%H,",",2) D S K %D,%M,%Y Q
```

- Sans resultat…

Cas 3: CBP

- Custom and Border Protection
- Application de validation des étrangers sur le sol américain
  - Plus de 4000 utilisateurs simultanés
  - 24h/24, 365 jours par an
  - Quelques minutes de downtime toutes les 7 semaines
  - La quantité de données est une information confidentielle
  - Mais elle se compte en tera-bytes

- COBOL/zOS/Datacom
  - Base de données pré-relationelle
  - Un souci industriel majeur
- Les contraintes sont énormes
  - Continuité de service
  - Criticité de l'application
  - Performance
- Le redéveloppement est en cours:
  - Java
  - Un modèle de données « amélioré »
  - Synchronisation bi-directionelle

Munis de ce viatique…

- Comment développer un système critique aujourd'hui?
- Identifier la durée de vie
  - Cathédrales et chateaux de sable
  - Mais les surprises sont fréquentes
- Simplifier l'architecture
  - TSPTHCW

- Découpe en couches
  - Présentation, logique applicative, persistance
  - Avec des durées de vie différentes
  - Et des interfaces suffisamment claires pour pouvoir remplacer un composant
- DIY
  - L'approche Raincode
  - Le précédent BoeingCalc
- Se reposer sur des standards
  - L'adéquation technique n'est (malheureusement) pas le seul critère !

No pain, no gain

- L'abus des standards est toxique
- Lorsque votre informatique est totalement générique, elle ne peut plus constituer une valeur ajoutée
  - Impensable de nos jours
  - Tous les domaines (transport, administration, énergie, distribution)
- Condamnés à s'écarter des standards
  - Toute l'astuce est de minimiser cet écart
  - Et le concentrer sur l'essentiel
  - Contre-exemple: Boeing Calc

Opposition

- Standards
- Peu d'abstraction
- Pérennité
- Limitation du risque
- Peu de valeur ajoutée

- Propriétaire
- Abstraction
- Précarité
- Risque
- Valeur ajoutée

RAINCODE

Voili, voila...