Fabien Appert

Jrat
« Java Runtime Analysis Toolkit »
Analyser pour optimiser

Plan de l'exposé

- Introduction
- Analyse de performance
- Optimisation
- Les outils
- Démonstration de Jrat
- Conclusion

Analyse de performance: pourquoi ?

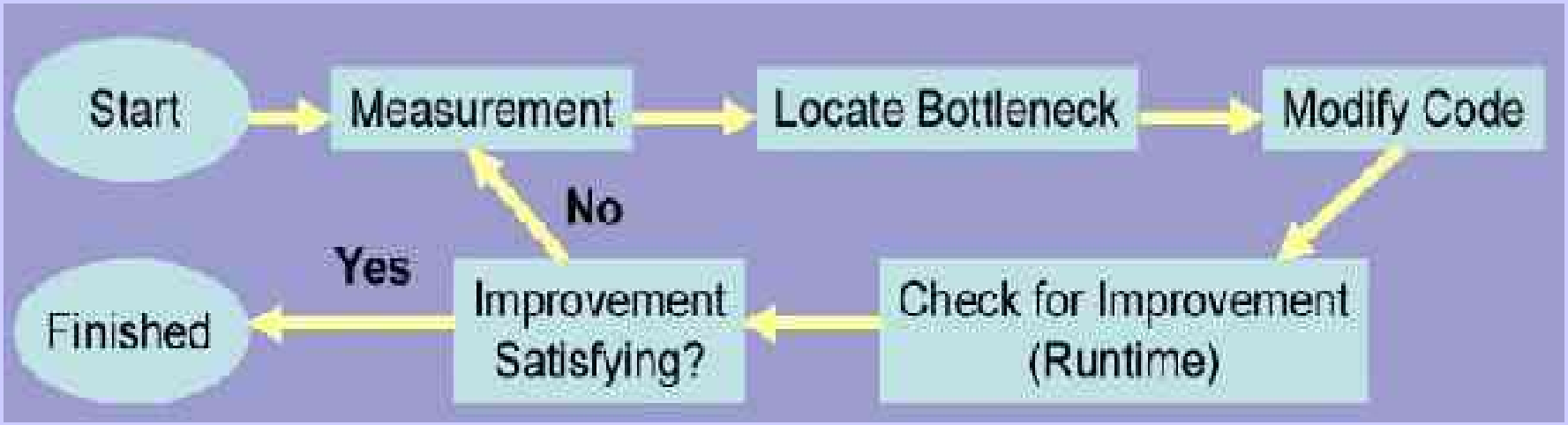
- Optimiser un programme ... mais quelle partie du code ?
- Choisir parmi plusieurs implémentation
- Vérifier et valider un comportement attendu

Analyse de performance : comment ?

- Par un code dont l'implémentation est terminée, et surtout testée !
- Sur une version optimisée par le compilateur
- Dans des conditions d'éxecution significatives

Optimisation

- Cycle d'optimisation :



Optimisation : comment ?

- Trouver les causes d'un goulot d'étranglement
- Trouver un meilleur algorithme
- Trouver des meilleures structures de données
- Verifier les limitations : problème des E/S
  - Accès disque... dommage
  - Accès mémoire... favoriser le cache

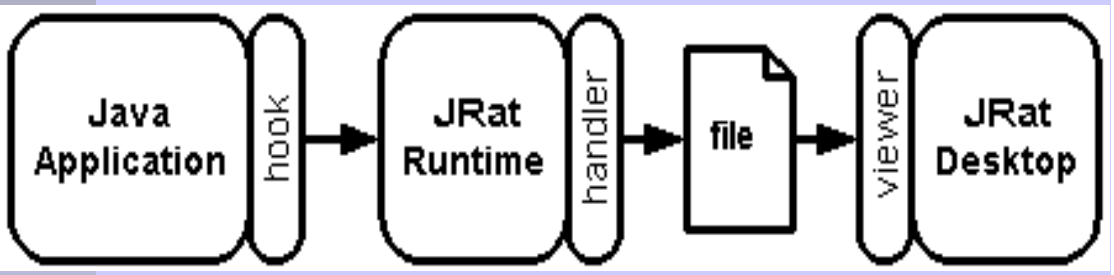
Optimisation : règles

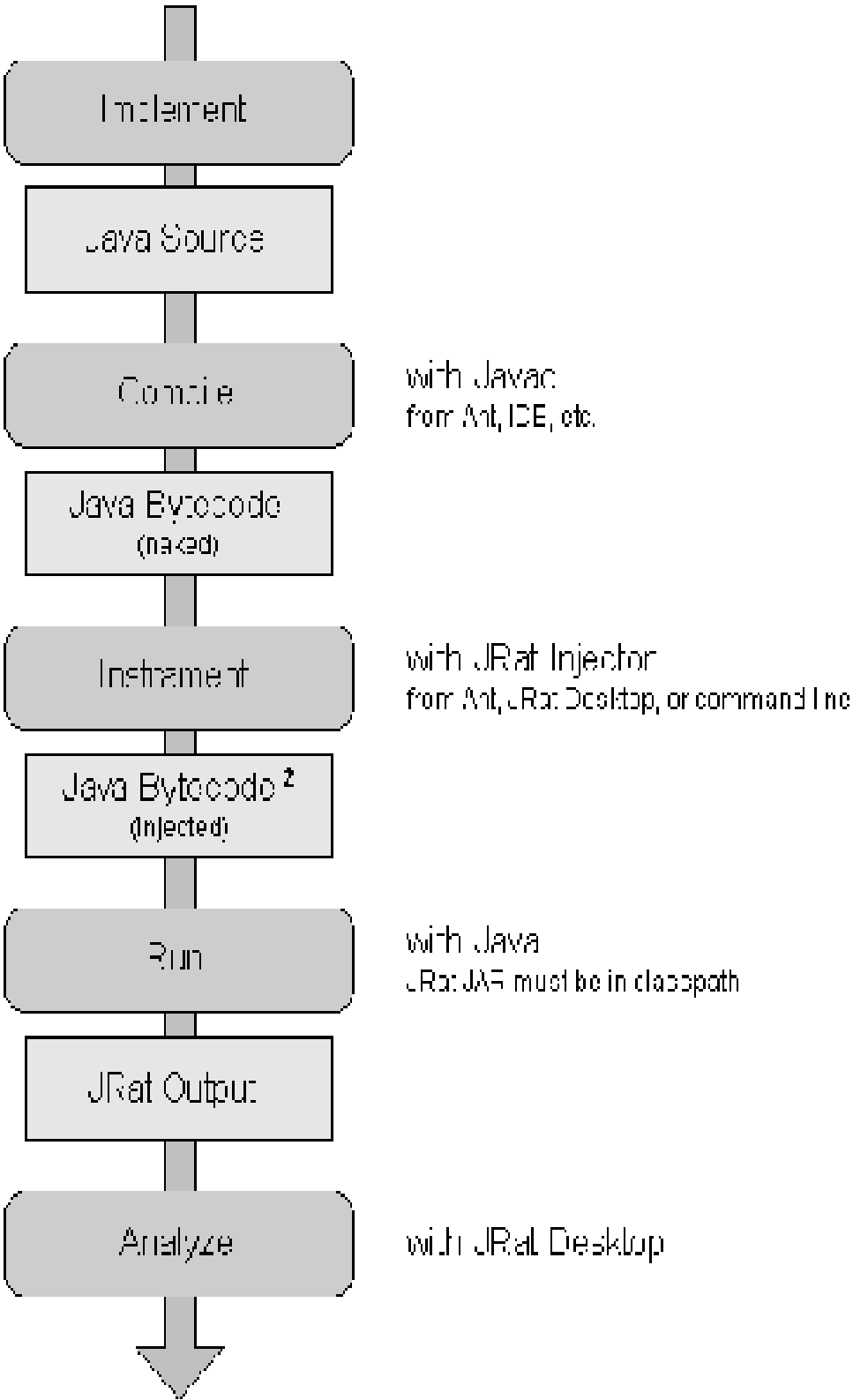
- Utiliser au maximum des bibliothèques déjà optimisées
- Utiliser des données et structures adaptées
- Penser aux optimisations du compilateurs
  - Variables *final*
  - Eviter les fonctions à rallonges, privilégier les fonctions courtes
  - etc...

Les outils

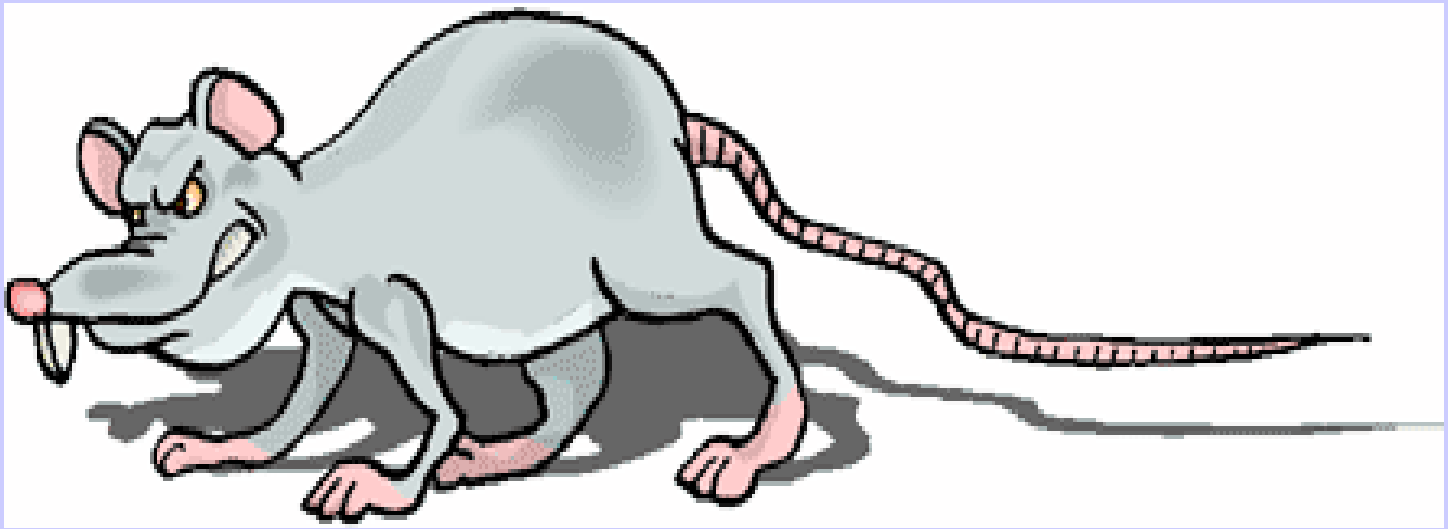
- L'interface JVMPI fourni par sun « Java Virtual Machine Profiling Interface »
  - API native
  - Nécessite le développement d'un agent
    - Exemple : hprof
- Suite payante : Borland Optimizeit Enterprise 6, Jprobe Suite, etc...
- OpenSource : JtreeProfiler, Jrat, etc...

Jrat : Architecture





Démonstration de Jrat ...



Exemple 1 : choisir un algorithme
Exemple 2 : tester un programme complexe

Fini !