It's Time To Get Functional

20071032 이건희
Context

• Intro
• Performance
• Popularity
• Network Effect

• Why Is It Time To Get Functional?
Intro
Architect

• You are the architect (or carpenter).
Tools

• Can you say “Oh, I have a fantastic multitester, so I don’t need a hammer!”?
Language Is A Tool

- 닭 잡는데 소 잡는 칼을 쓰지 마라
- To take a sledgehammer to crack a nut.
- It is very important to use proper tools.
Performance

• Performance critical applications
  – Game, scientific calculation, database

• Not so critical applications
  – Wordprocessor, email client, messenger
Performance

• Not just runtime performance
• Because we are the architect

• Developing performance
• Runtime performance
  – Stability, Speed, Space
Developing Performance

• RAD tools
  – Rapid Application Development
• Delphi, Visual Basic, Visual C#, ...

• Very easy GUI implementations
• But limited abilities
Developing Performance

• Python, Perl, Ruby, ...

• Scripting languages

• A lot of libraries (modules) out there

• Hackers love these
Runtime Performance

- Speed (general consensus)
- C, C++
- Haskell, Erlang, Ocaml, ...
- Java
- Assembly (King)
Just A Benchmark

Benchmark Results
excluding Python & Python/Psyco

Seconds

Language

Visual C++ .NET
Visual C# .NET
C GCC 3.3.1
Visual Basic .NET
Visual J# .NET
Java 1.3.1
Java 1.4.2

i/o
trig
long math
do double math
int math
Runtime Performance

• Space

• Graphic / Audio / Video editing

• Generally not dependent on language
Runtime Performance

• Stability

• Where Functional Programming truly shines

• Type safety
Runtime Performance

• However, as Prof. gla mentioned,

• Functional programming has potential to be the best racer!

• Concurrent programming
Bad News
Popularity

- Java - 19.537%
- C - 16.128%
- C++ - 11.068%
  - (May, 2009)
Popularity

- LISP/Scheme - 0.465%
- Erlang - 0.190%
- Haskell - 0.172%
- ML - 0.141%
What's Wrong?

- We all know that SML is way better than C
- Does technical superiority not matter?
- How is this possible?
Network Effect

• Think about it :: When you find an error message, what do you do?

• The answer :=

  Google™
Network Effect

- If there are already millions of users out there, you can find your answer more easily.

- This is a very common phenomenon.
Network Effect

- Ex) CPAN (www.cpan.org)
- Comprehensive Perl Archive Network
- You can find 15856 modules (libraries)
- FAQ, Docs, QnAs, Modules, etc...
Network Effect

• I insist that in this level of popularity, functional programming has no vision in commercial programming.

• But, how about in 2020?
Future

• Surely there will be a huge improvement!

• That's one of the reason why we should learn functional programming

• We can prepare for the future
Why?

• Why there will be an improvement?
• We should see a little physics lecture
Multicore

• You heard about dual-core, quad-core, octa-core, ...

• Why?
Clock Rate

- Air cooling = max. 3.7GHz
- Water cooling = max. 4.5GHz
- Liquid Nitrogen cooling = max. 7.35GHz

- No more
- These are the limits
Physical Barriers

- Speed of electrons $(2c/3)$
- Overheating
- Microwave emission
- Quantum effects
Answer

- Bio-computing
- Quantum-computer
- Feltier cooler
- Multicore paradigm
Multicore!
Not A Perfect Answer

• There is a formula called Amdahl's Law.

\[
\frac{1}{(1 - P)} + \frac{P}{N}
\]
Data Hazard

• And another problem :: Data hazard

• Both cores share same cache and bus

• What if these two cores alter the same data concurrently?
Stability

- In multicore environment, stability is the key performance
- More stable and pure (in the sense of immutable variables) program means more parallel program
Not Enough!

• Showing off the superiority of FP is not enough!

• Criticizing C and C++ and other languages doesn’t help FP
Strengths

• To get the maximum potential of FP
• We should know the strengths and weaknesses
• Improve step by step
Application 1
Application 2
Application 3
Not Good At

• Game Programming
  – Haskell, Yampa, and Clean barely support this by using “impure” methods
• GUI Programming
  – Haskell has a good library
  – F# uses .net framework
  – But, still hard
Which One?

- Haskell
  - Open source, extensible, has good libraries

- F#
  - Supported by MS
  - .net framework
Conclusion

• Prepare for the future!
• Learning the usage of new tool doesn't hurt you.
• However, don't underestimate other programming languages.
Question