

# The Future of Software is a Blend, But of What?

Bob Sutor

VP, Standards and Open Source, IBM

[sutor.com/blog](http://sutor.com/blog)

July 27, 2006



# Twelve assumptions about the future: #1 - 3

- The world is not going to revert suddenly to the previous model of nearly all proprietary software.
- Neither is the world going to be completely open source any time soon.
- Areas that are now or should get commoditized will move largely to open source in the next 5 years.

# Twelve assumptions about the future: #4 - 6

- We will have “open source pretenders,” vendors who say they get open source and will move in that direction, when their intention is very much to the contrary.
  - Trust no one who doesn't prove himself by honest participation, but do factor in some slack due to resource constraints and currently profitable business models.
- For many users, the cost of licensing will not be the main reason to shift to open source.
- For many users, particularly governments, the cost of licensing software will be the major issue and will force a significant shift to open source over the next 3 years.

# Twelve assumptions about the future: #7 - 8

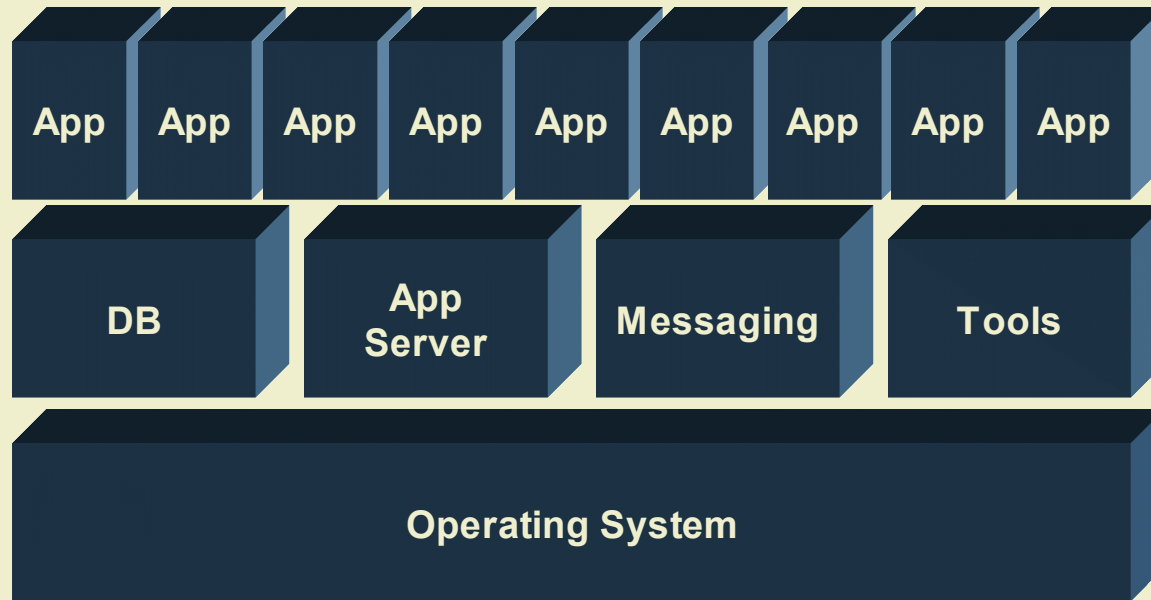
- A nontrivial number of developers and salespeople of proprietary software do and will get seriously annoyed when you suggest they move to an open source business model.
  - Conspiracy theories abound.
- As people better understand what constitutes a real open standard, open source implementations of such standards will and should increase.
  - These will further prevent “embrace and extend” strategies, though it might not forestall the “we’ll just do another standard” tactic.

# Twelve assumptions about the future: #9 - 12

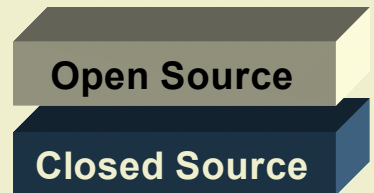
- We will continue to see the opponents of “open” get more clever and subtle, making the counter-arguments more difficult, in part because they are adaptations of the arguments that are pro open.
- Enterprise software strategies that do not include Linux and Eclipse will get marginalized.
- While we may be past the “open source as ideology” phase, that doesn’t mean that supporters will or should stop or decelerate their code development and evangelism.
- Mixed middleware stacks will be dominant within 3 years, as will mixed desktop stacks in 5 years.

# In the Beginning There Was ...

1

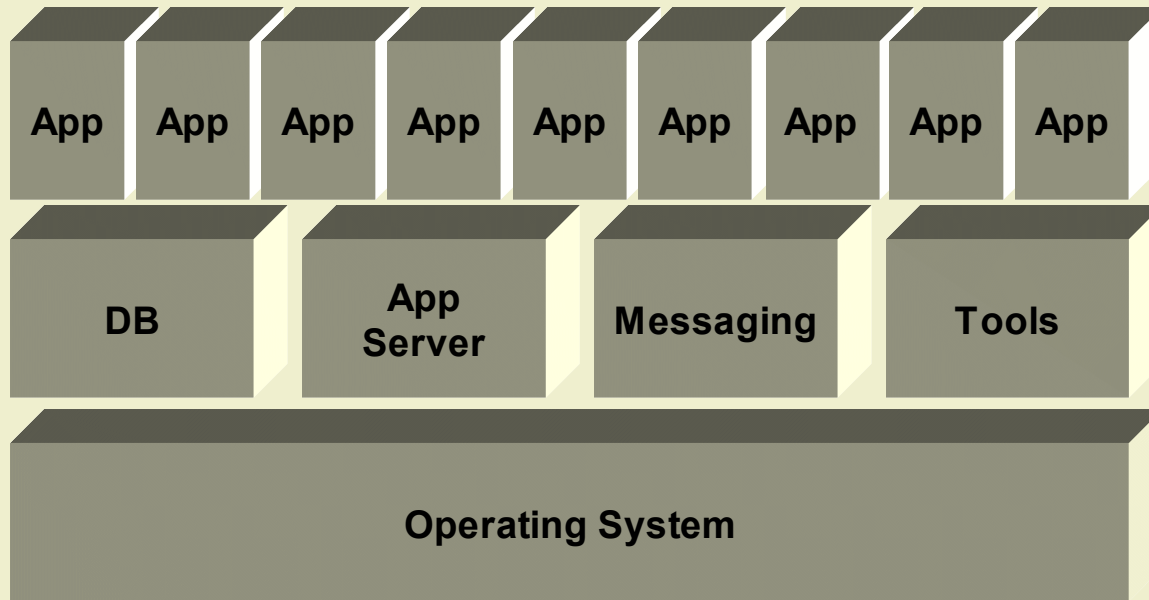


Key

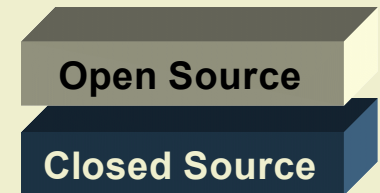


# Perhaps Eventually There Will Be ...

2

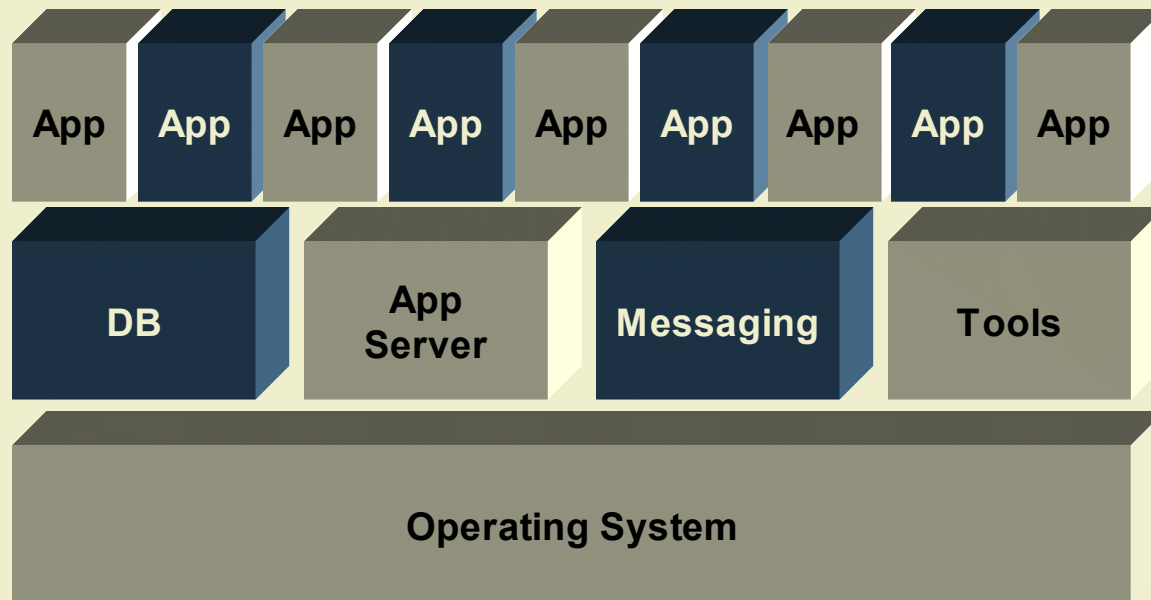


Key

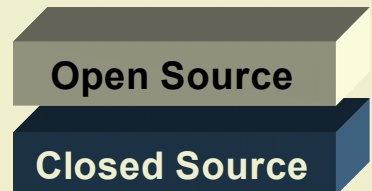


# But in the Meanwhile We Have ...

3

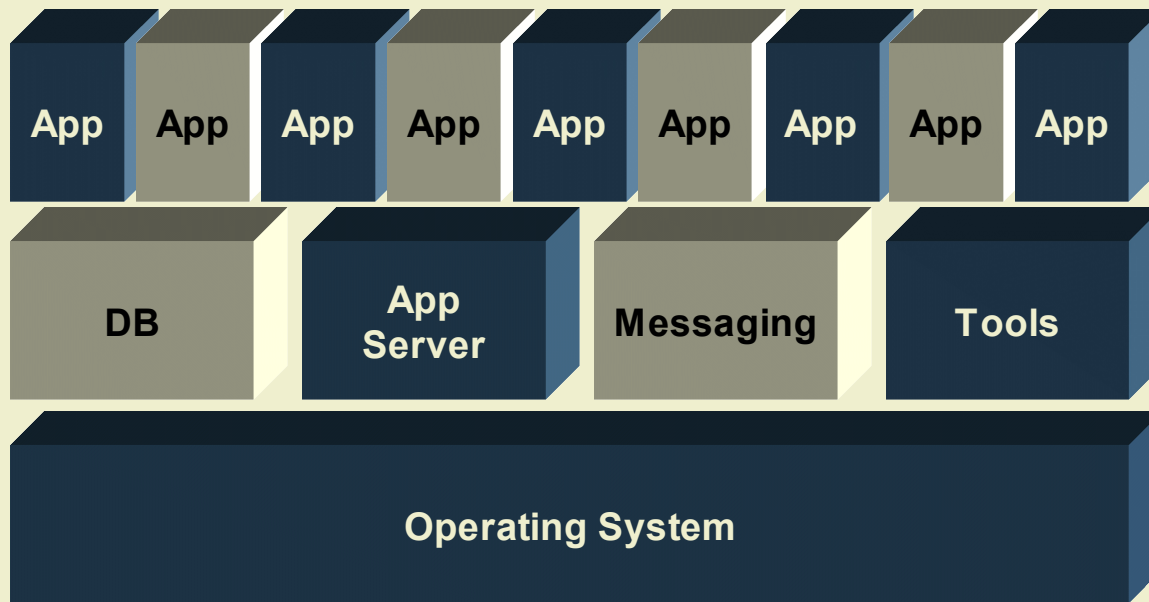


Key

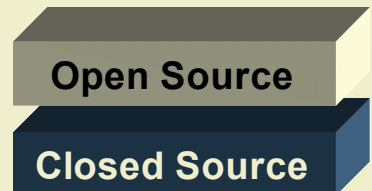


and ...

4

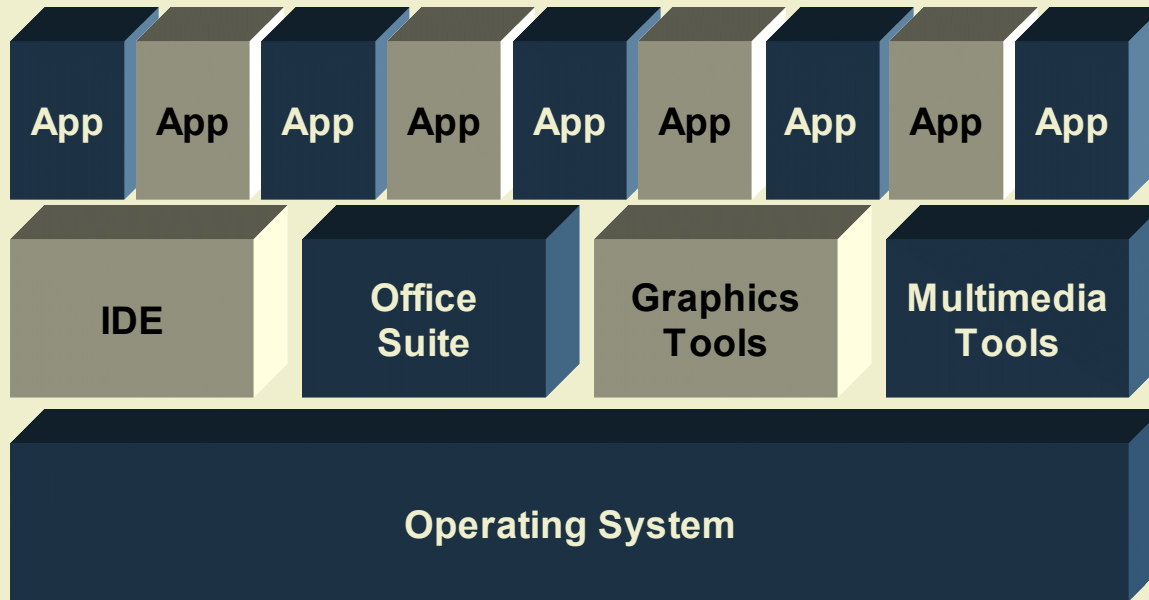


Key

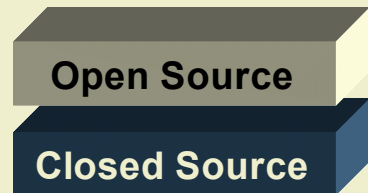


# and ... (You Get the Idea)

4'



Key



# Ask yourself these questions

- As a software developer, what is the best combination of proprietary and open source tools and run-time platforms that will let me deliver the highest value solution to my customers as quickly and economically as possible?
- As an IT-savvy business person, what is the right mix of proprietary and open source software that will allow me to have the most economical, scalable, secure, best performing, and future-proof IT configuration?

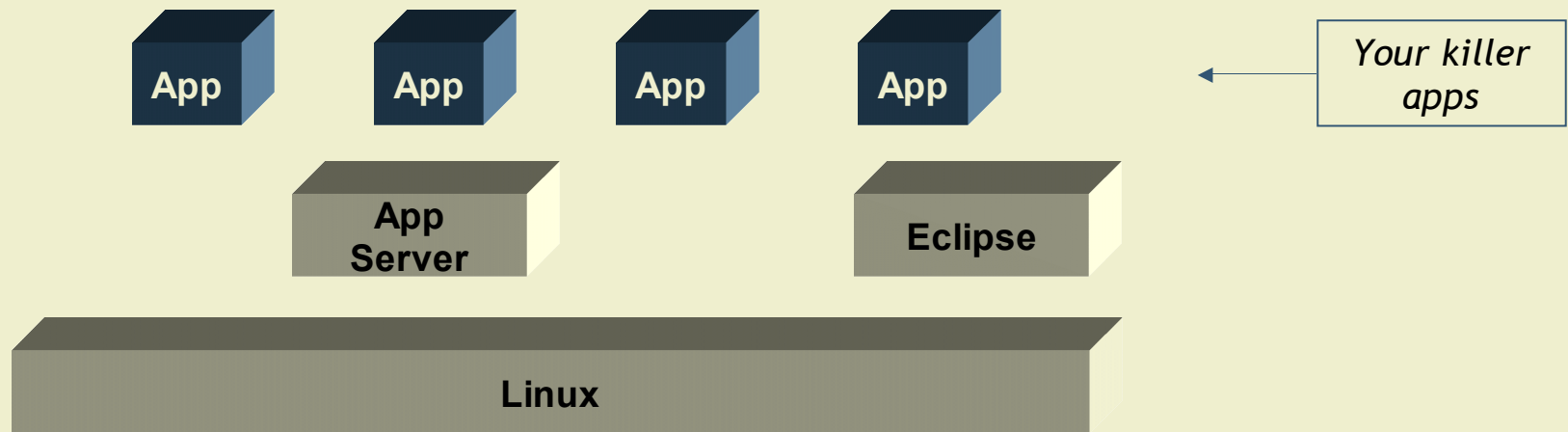


# How Do You Choose Among These?

- Dogmatically choose #1 (all closed) or #2 (all open)?
- Sprinkle open source on top of a closed foundation (#3)?
- Do as much open as you are comfortable with, but fill in with closed software (#4)?
- Build a long range plan to go from all closed to all open?
- Let a vendor/distributor decide for you?
- Let a stack builder decide for you?

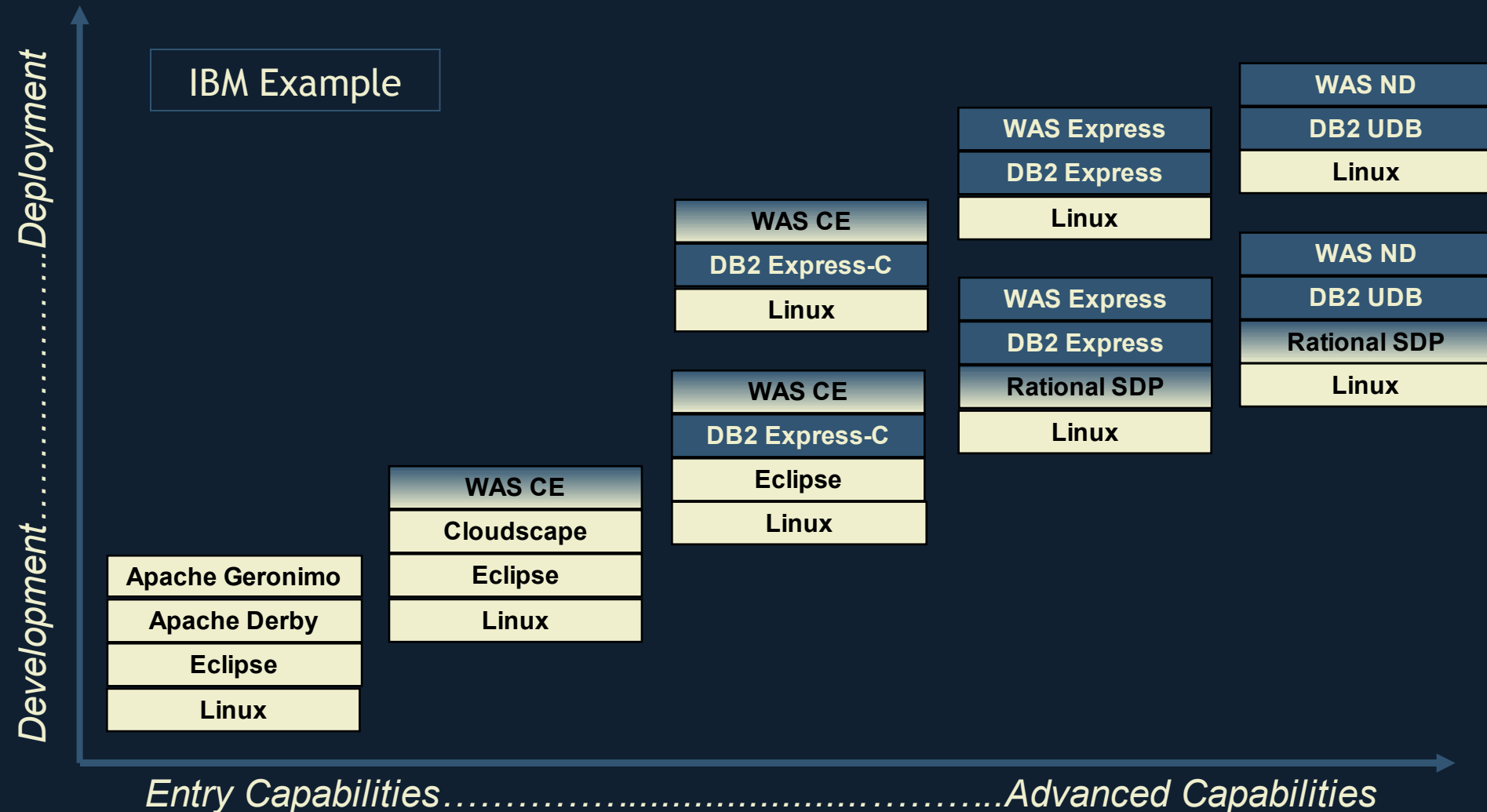
# Building Your Own Stack

- Start with Tomcat and Eclipse on Linux because it's quick and easy



- But what if you need clustering, web services, messaging, and security?

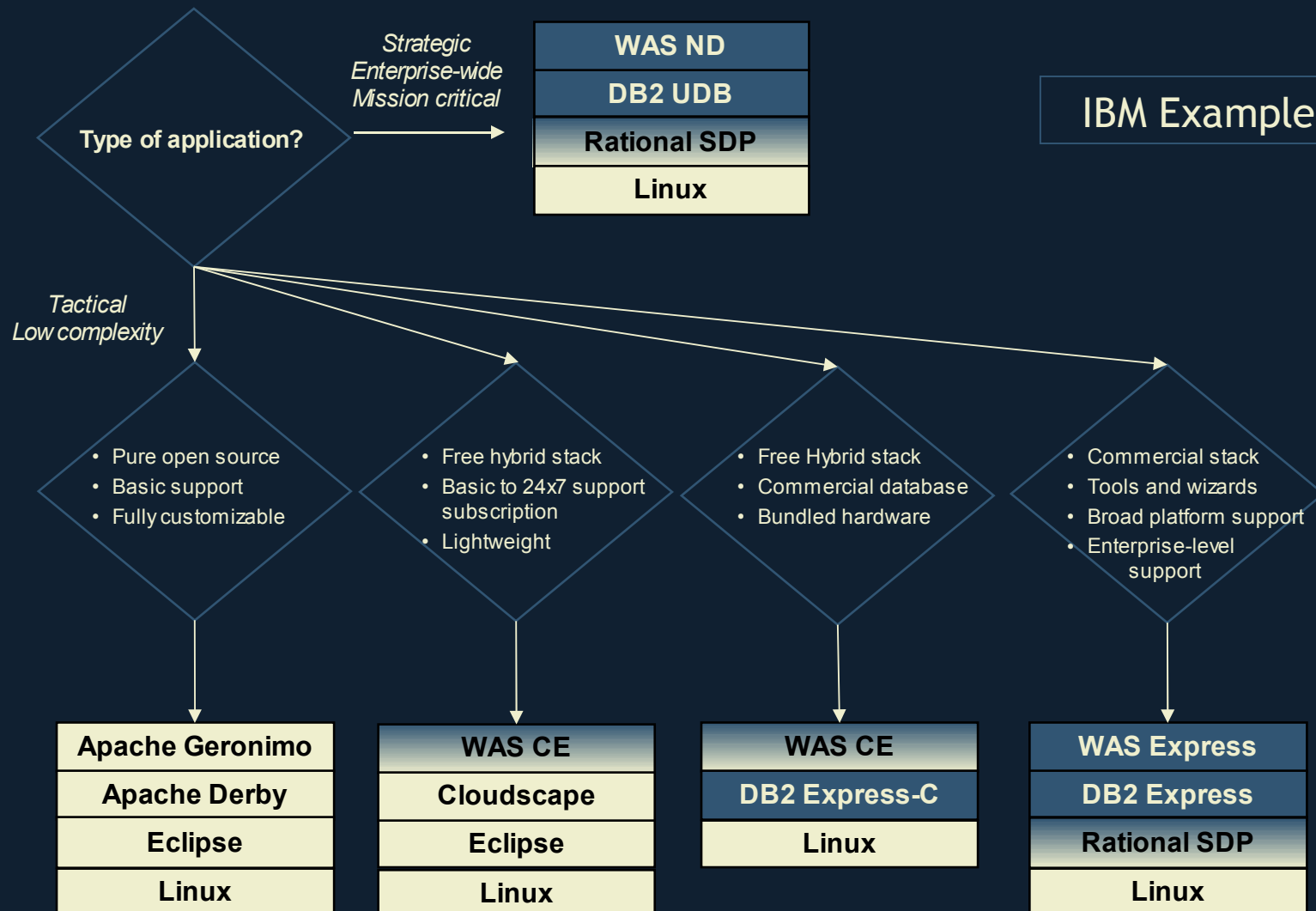
# A Family of Stacks for Linux



Sutor: The Future of Software is a Blend, But of What?



# How Do You Decide What's Good Enough?



# Recommendations

- Follow the money, er, standards.
- Don't just clone the market leader, change the market category.
- Think workflow, not point products.
- Beware lock-in plays above the commoditizing layer.
- Choose the optimal stack for yourself based on
  - Project complexity (strategic, corporate-wide vs. tactical, departmental)
  - Level of support (correlated with skill sets you have in-house)
  - Strong preference for pure open source
  - Architecture for third-party plugins
  - Commercial characteristics such as broad platform support, documentation, tools
  - Structure of budget

# You get to decide

- Ultimately, do whatever gives you the most *choice* of and *control* over
  - architectural configuration
  - software used
  - software providers and maintainers
  - service and solution providers
  - underlying platforms (OSs, hardware, devices)
  - financial options
  - use of community developed open standards
  - factors for managing risk
  - confidence-building elements for the future
- There are many companies that can help you make the decision.