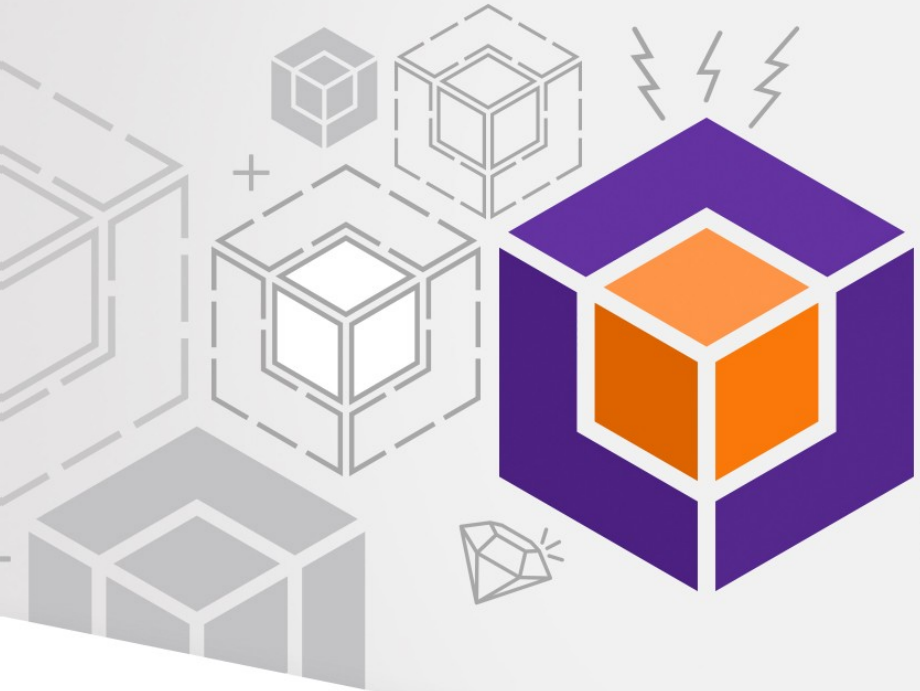
YottaDB

August 30, 2020



M[UMPS] (the language) is (nowhere near) Dead; Long Live M[UMPS] (the database)!

A Way for VistA to “Go” Forward

YottaDB® – https://yottadb.com

- A mature, high performance, hierarchical key-value NoSQL database whose code base scales up to mission-critical applications like large real-time core-banking and electronic health records, and also scales down to run on platforms like the Raspberry Pi Zero, as well as everything in-between.
- *Rock Solid. Lightning Fast. Secure. Pick any three.*

Cities

YottaDB

- Cities are never designed to become obsolete
- Julius Caeser knew London, Paris, and Rome
  - But he wouldn't recognize any of them today
- Successful cities evolve
  - Those that can't adapt, die, e.g., Copán
  - Continuous evolution inevitably mixes technologies
    - e.g., Fibre-optic networking with medieval sewers

VistA

YottaDB

- No planned obsolescence
  - But regular attempts to kill it by starvation
    - [Not unlike laying siege to a city]
- Still recognizable to its original developers
- Adaptation / evolution still in doubt
  - Separating political from technical issues not simple
  - Technology exists to evolve; is the political will there?

What Defines a City?

YottaDB

- Location and people
- Individuals, buildings, roads, railways, etc. are transient

What Defines VistA?

YottaDB

- Data about people
- Individuals, interfaces, logic, etc. are transient
  - Even programming languages like M[UMPS]

What Makes M[UMPS] Unique?

YottaDB

- Tight binding of database to language
- Other features are powerful, but secondary

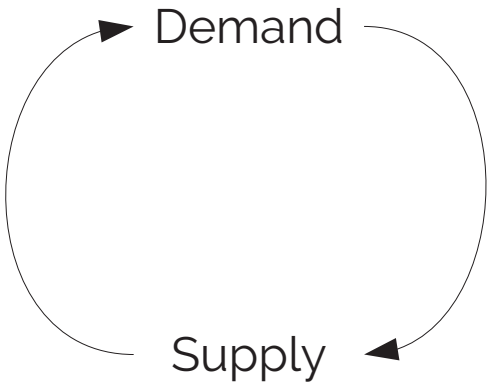
M[UMPS] Challenges

YottaDB

- Insular community
- Frozen evolution
- Even the biggest vendor won't call it by name
- It's not something the next generation of programmers wants on their resumes
  - Whatever it is called
  - *And we're too small a community to change that*

Programmer Supply & Demand

YottaDB

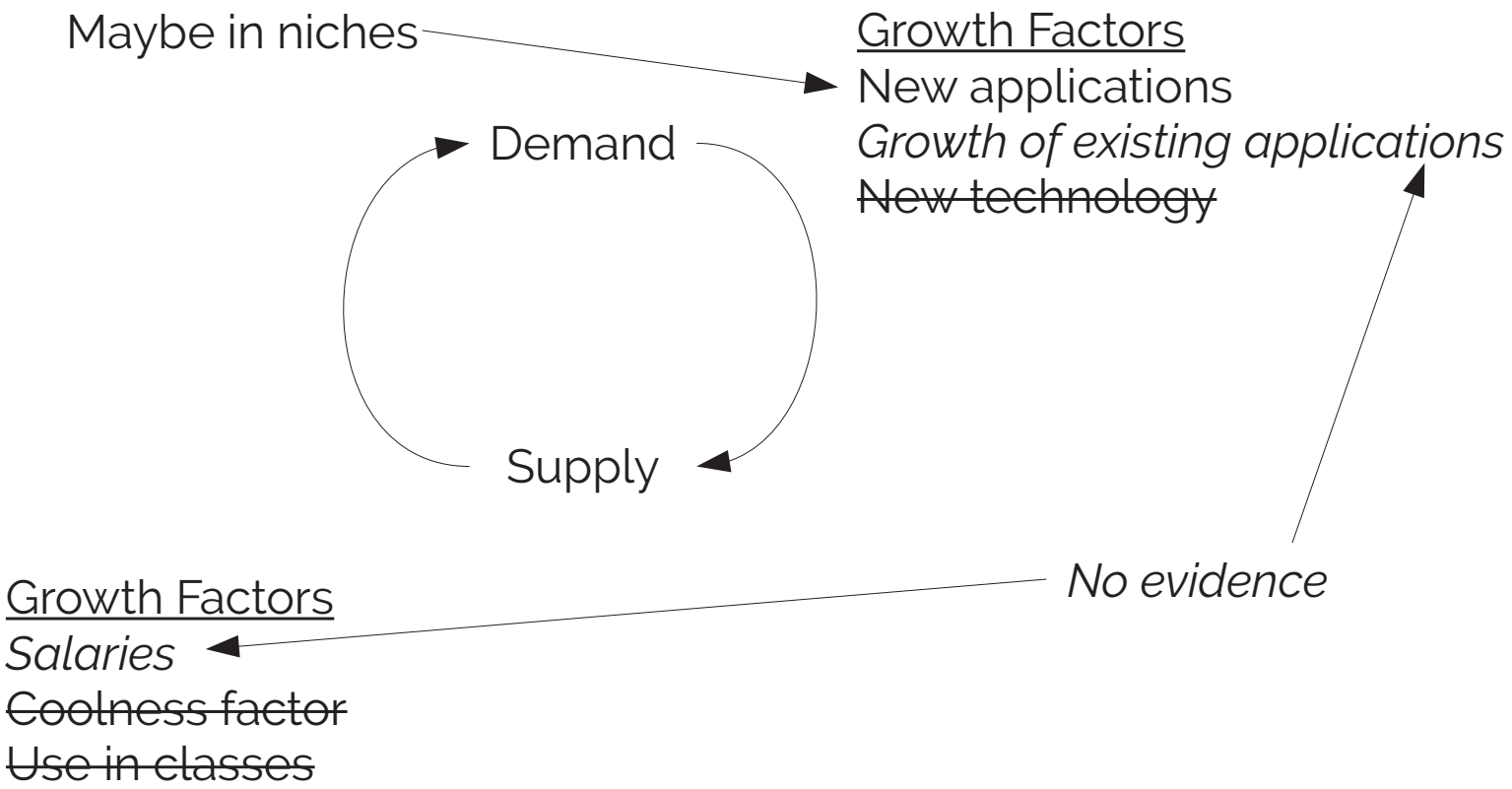


Growth Factors
New applications
Growth of existing applications
New technology

Growth Factors
Salaries
Coolness factor
Use in classes

M[UMPS] Programmer Supply & Demand

YottaDB



How Do Cities Evolve?

YottaDB

- Build on the old and embrace the new
  - With selective, geographically limited redevelopment from time to time
- Coexistence

How Can VistA Evolve?

YottaDB

- Provide new ways to acccess and use the data
  - With selective, redevelopment from time to time of limited functional areas
- Coexistence
  - Design new functionality to benefit from old functionality and to allow old functionality to benefit from it

YottaDB Approach

- Build on what works well
- Accommodate what's new

Public domain from Wikimedia Commons

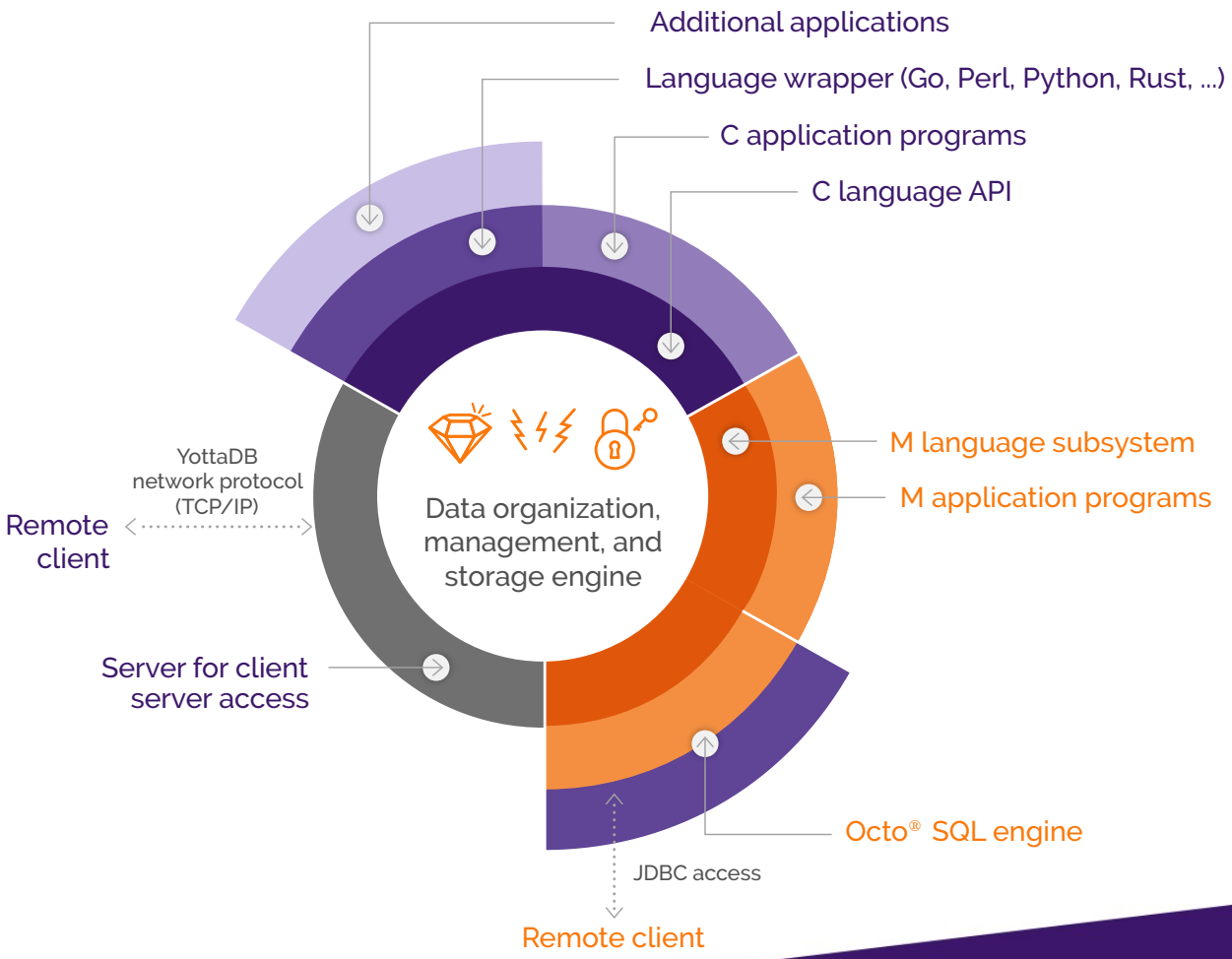
By GT1976 [CC BY-SA 4.0 (https://creativecommons.org/licenses/by-sa/4.0)], from Wikimedia Commons

Photos are almost 100 years apart

The YottaDB View

- The diamond is the database
- The language is what it is
  - Like anchovies on pizza, you either love it or hate it
- Solution: language agnostic database
  - Take nothing away from M[UMPS], the language
  - Make M[UMPS], the database, accessible from other languages

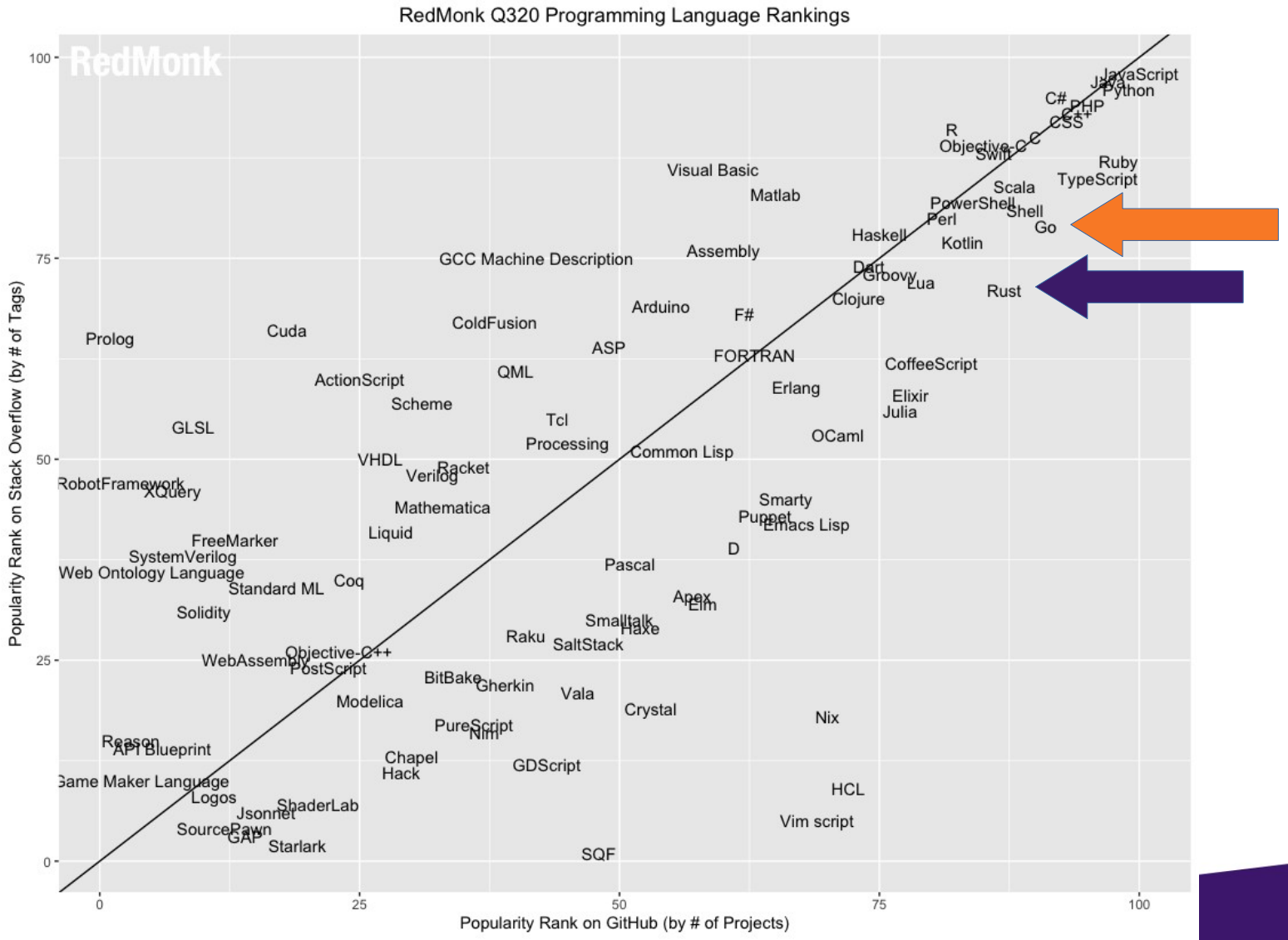
YottaDB Data-Centric Architecture



Go

YottaDB

- Developed by Google and used internally
  - https://golang.org
- Released as free / open source
  - Active user community
  - Growing popularity
- YottaDB's first wrapper
  - https://yottadb.com/yottadb-go-wrapper/


RedMonk Q320 Programming Language Rankings
RedMonk
Popularity Rank on Stack Overflow (by # of Tags)
Popularity Rank on GitHub (by # of Projects)
0
25
50
75
100
JavaScript
Java
Python
C#
PHP
C++
CSS
R
Objective-C
C
Swift
Ruby
TypeScript
Scala
PowerShell
Shell
Perl
Go
Haskell
Kotlin
Visual Basic
Matlab
Assembly
GCC Machine Description
Dart
Groovy
Lua
Rust
Clojure
Arduino
F#
ColdFusion
Cuda
Prolog
ASP
FORTRAN
CoffeeScript
QML
ActionScript
Erlang
Elixir
Julia
Scheme
Tcl
GLSL
Processing
OCaml
Common Lisp
VHDL
Verilog
Racket
RobotFramework
XQuery
Mathematica
Smarty
Puppet
Emacs Lisp
Liquid
FreeMarker
SystemVerilog
D
Pascal
Web Ontology Language
Standard ML
Coq
Apex
Elm
Solidity
Smalltalk
Haxe
Raku
SaltStack
Objective-C++
WebAssembly
PostScript
BitBake
Gherkin
Vala
Modelica
Crystal
Nix
PureScript
Nim
Reason
API Blueprint
Chapel
Hack
GDScript
Game Maker Language
HCL
Logos
ShaderLab
Jsonnet
Vim script
SourceRawn
GAP
Starlark
SQF

Go co-designer Rob Pike:

YottaDB

- https://talks.golang.org/2012/splash.article

  … Go's design considerations include *rigorous dependency management*, the adaptability of software architecture as systems grow, and *robustness across the boundaries between components*. … Go is a *compiled, concurrent, garbage-collected, statically typed* language developed at Google. It is an open source project: Google *imports the public repository* rather than the other way around. …. Go is *efficient, scalable, and productive*.

YottaDB Go Wrapper

- CallMT, Data, Delete, DeleteExcl, Incr, Lock, LockDecr, LockIncr, NodeNext, NodePrev, SetVal, SubNext, SubPrev, TP, Val
- Two variants: Easy API & Simple API
  - Developed in consultation with and intuitive to Go programmers
- Full set of utility functions

Simulated Balance Transfers – M

```
baltrans(fromacct,toacct,amount)
      new frombalance                          ; local variables used in this routine
      tstart ()                                ; no local variables to be restored on restart
      set frombalance=^balance(fromacct)       ; cache global in a local for performance
      set:amount>frombalance $ecode=",U123,"   ; check for sufficient funds; raise error if not
      set ^balance(fromacct)=frombalance-amount
      set ^balance(toacct)=^balance(toacct)+amount
      tcommit
      quit
```

Simulated Balance Transfers – Go

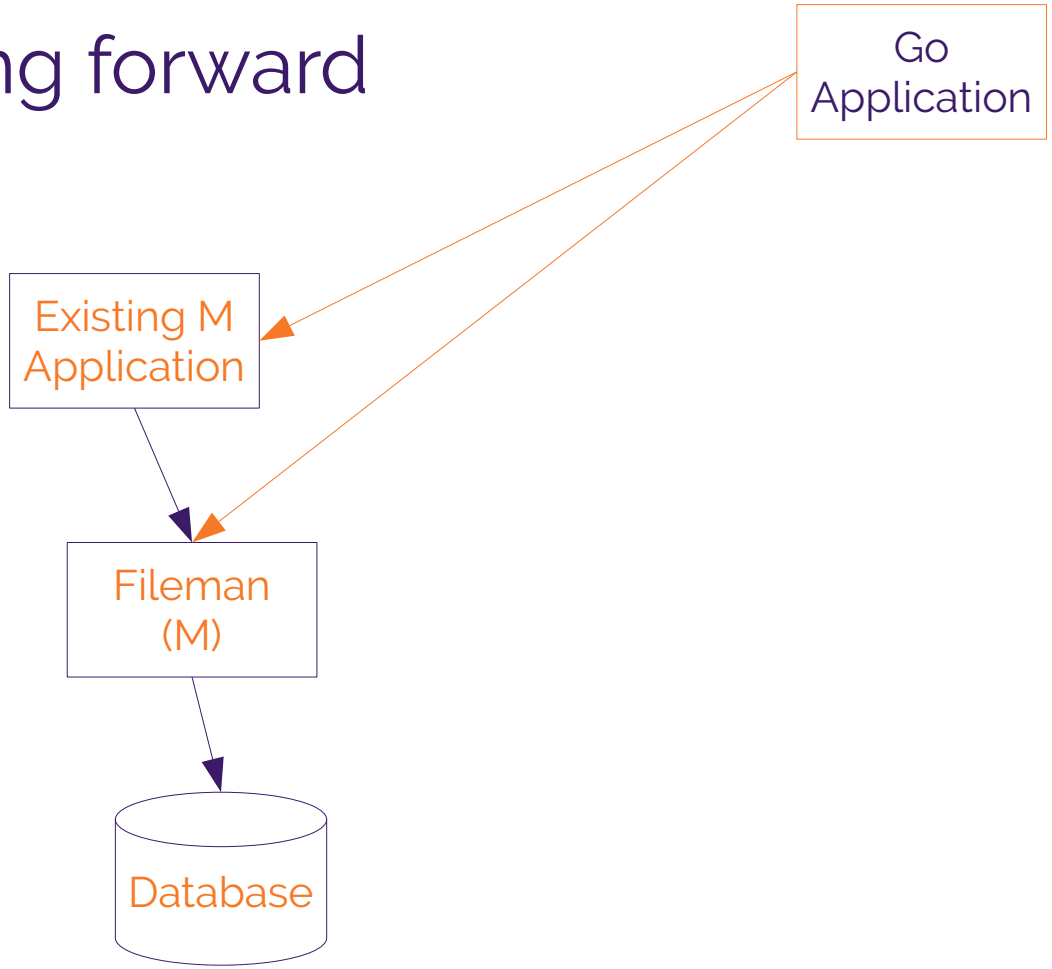
```
func baltrans(fromacct string, toacct string, amount int) {
  var errstr yottadb.BufferT
  errstr.Alloc(1024)
  yottadb.TpE(yottadb.NOTTP, &errstr, func(tptoken uint64, errstr *yottadb.BufferT) int32 {
    from_balance_s, _ := yottadb.ValE(tptoken, errstr, "^balance", []string{fromacct})
    from_balance, _ := strconv.Atoi(from_balance_s)
    if amount > from_balance {
      return yottadb.YDB_TP_ROLLBACK
    }
    from_balance -= amount
    yottadb.SetValE(tptoken, errstr, fmt.Sprintf("%d", from_balance), "^balance", []string{fromacct})
    to_balance_s, _ := yottadb.ValE(tptoken, errstr, "^balance", []string{toacct})
    to_balance, _ := strconv.Atoi(to_balance_s)
    to_balance += amount
    yottadb.SetValE(tptoken, errstr, fmt.Sprintf("%d", to_balance), "^balance", []string{toacct})
    return yottadb.YDB_OK
  }, "", nil);
```

Moving Forward

YottaDB

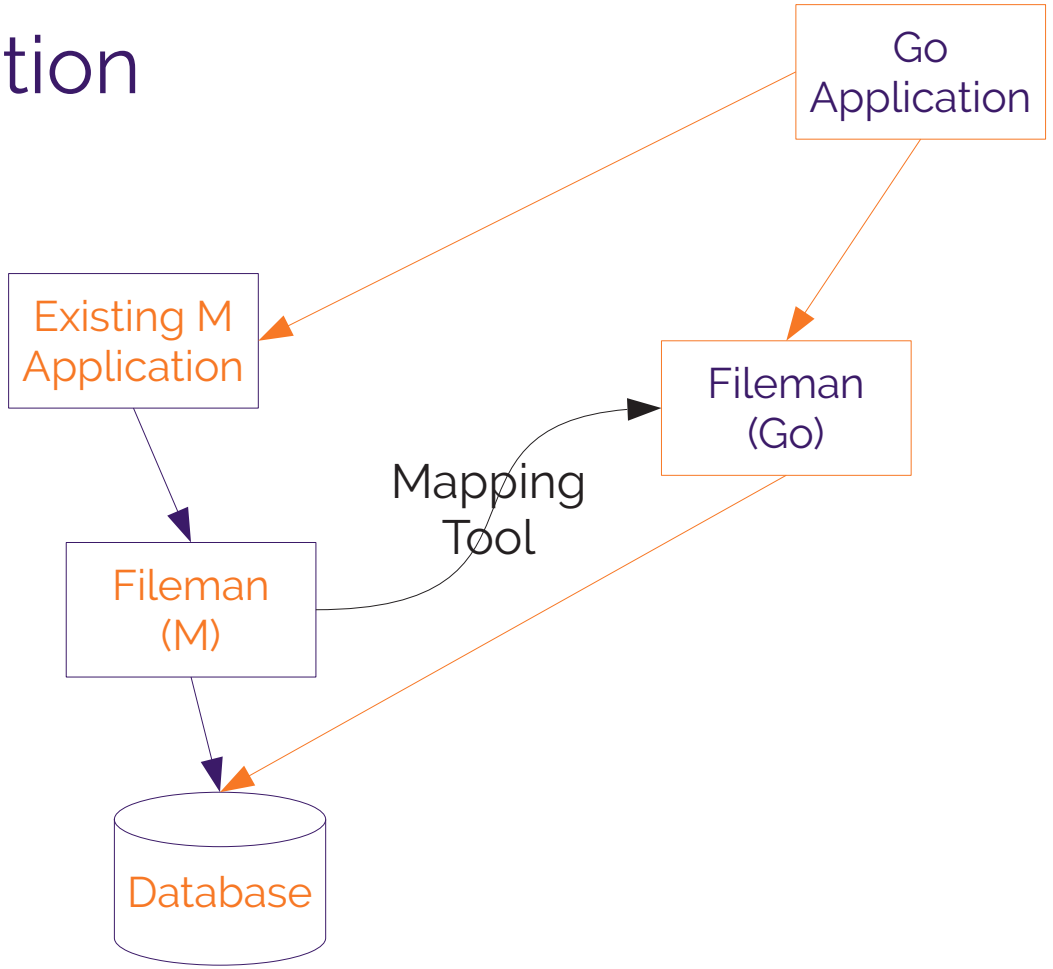
- Not necessarily one correct way forward
  - Doubtful that M[UMPS] can evolve its way to world domination
  - Go is promising – performance, garbage collection, popularity, free / open source, use by Google, etc.
- To not evolve is to stagnate, and fade into irrelevance

“Go”ing forward



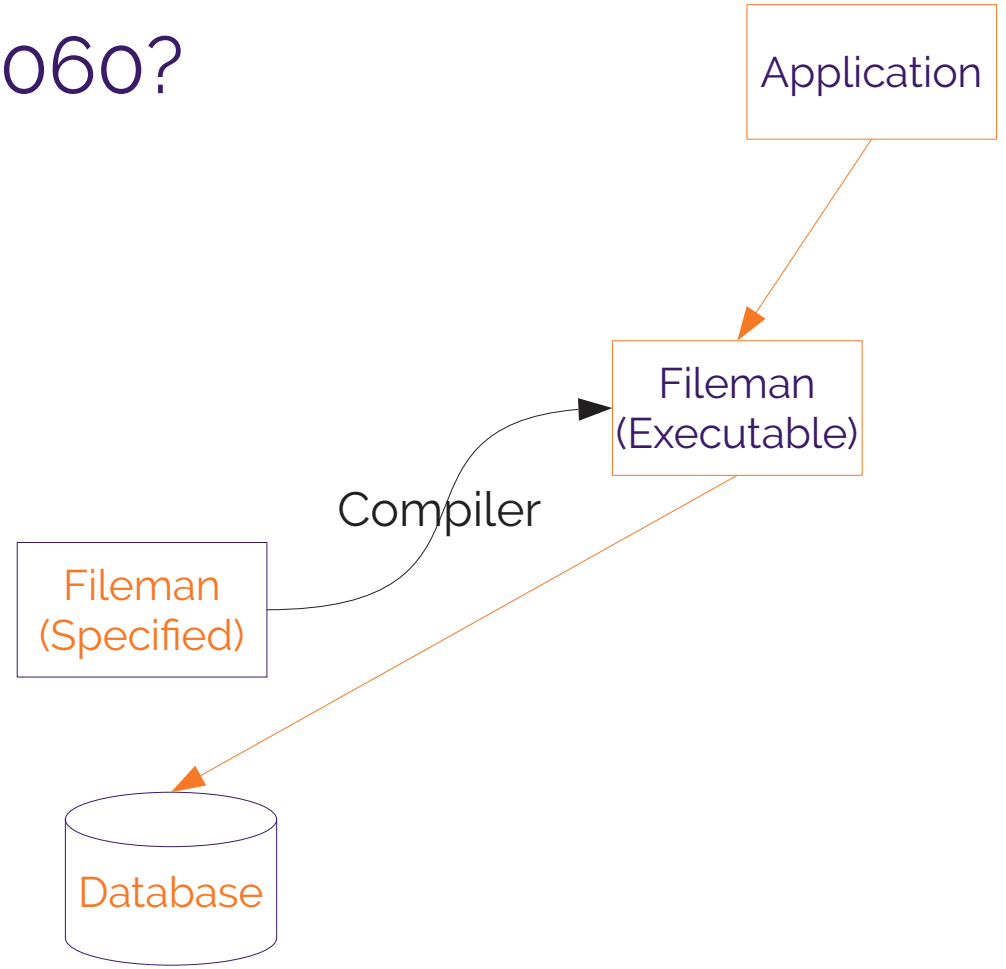
YottaDB

Evolution



YottaDB

VistA 2060?


Application
Fileman
(Executable)
Compiler
Fileman
(Specified)
Database

YottaDB

Thank You!

K.S. Bhaskar
bhaskar@yottadb.com