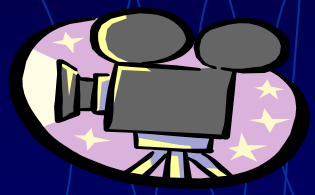
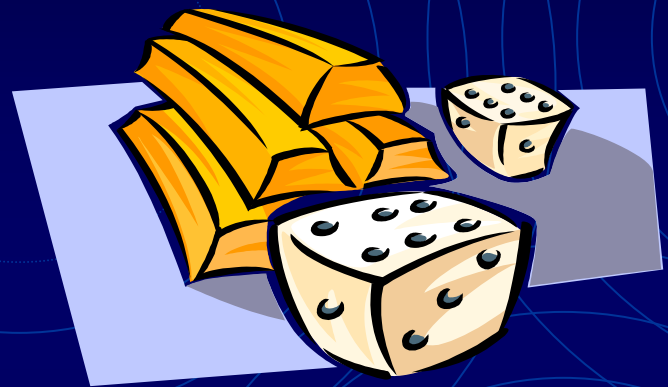
# Behind the Scenes



- or -

Where does the  go?

If Las Vegas sounds too tame for you,

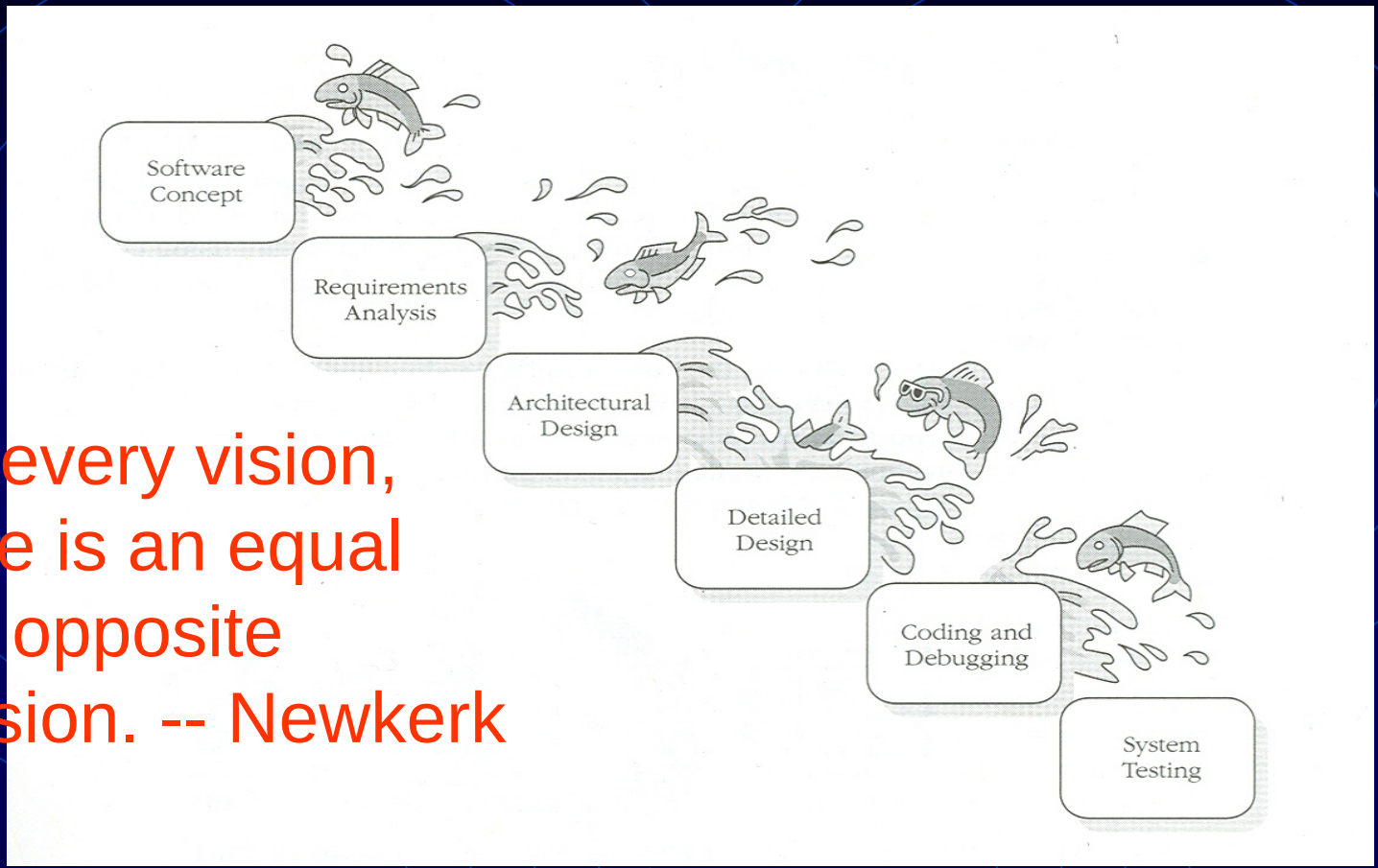


software might be just the right gamble.

- Steve McConnell

Loss of personnel. Quality problems. Optimistic scheduling. Software project risk management takes more time than expected. Feature creep. Inadequate design. Silver-bullet syndrome. Shifting requirements. Project is larger than estimated. Schedule omits necessary tasks. Strict requirements for compatibility with existing system require more testing than expected. Target date is moved up with no adjustment in project scope or resources. Development tools are not in place on time. Budget cuts upset project plans. End user insists on new requirements. Third-party takes longer than expected. Unfamiliar hardware environment requires more training time. Dependence on an unproven technology. Customer communication time is slower than expected. Operation in an unproven environment causes unforeseen problems. Product depends on government regulations or standards, which change unexpectedly. Not enough personnel for the project. Amount of paperwork slows progress more than expected. Tool little formality. Too much formality.

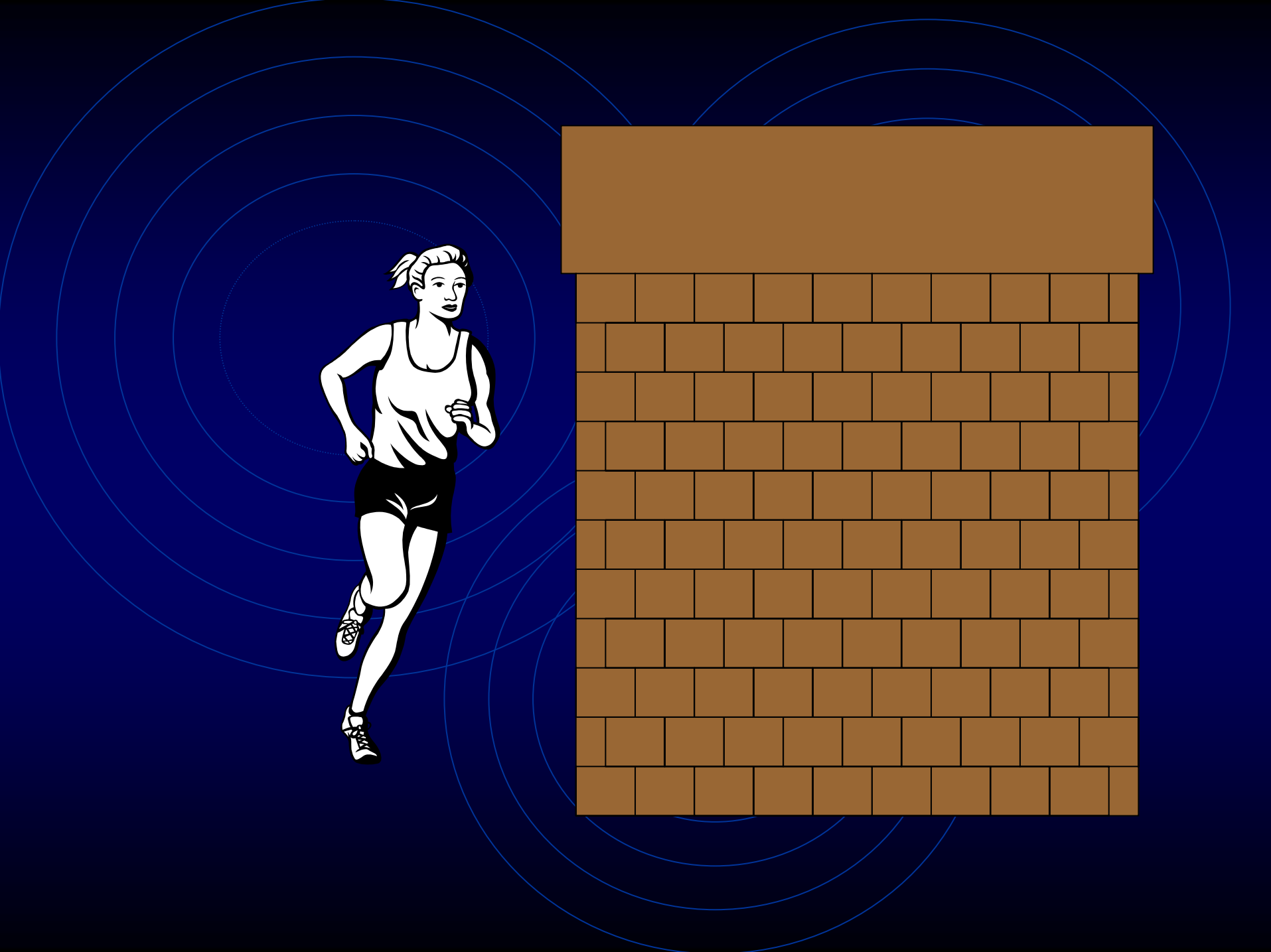
For every vision, there is an equal and opposite revision. -- Newkerk



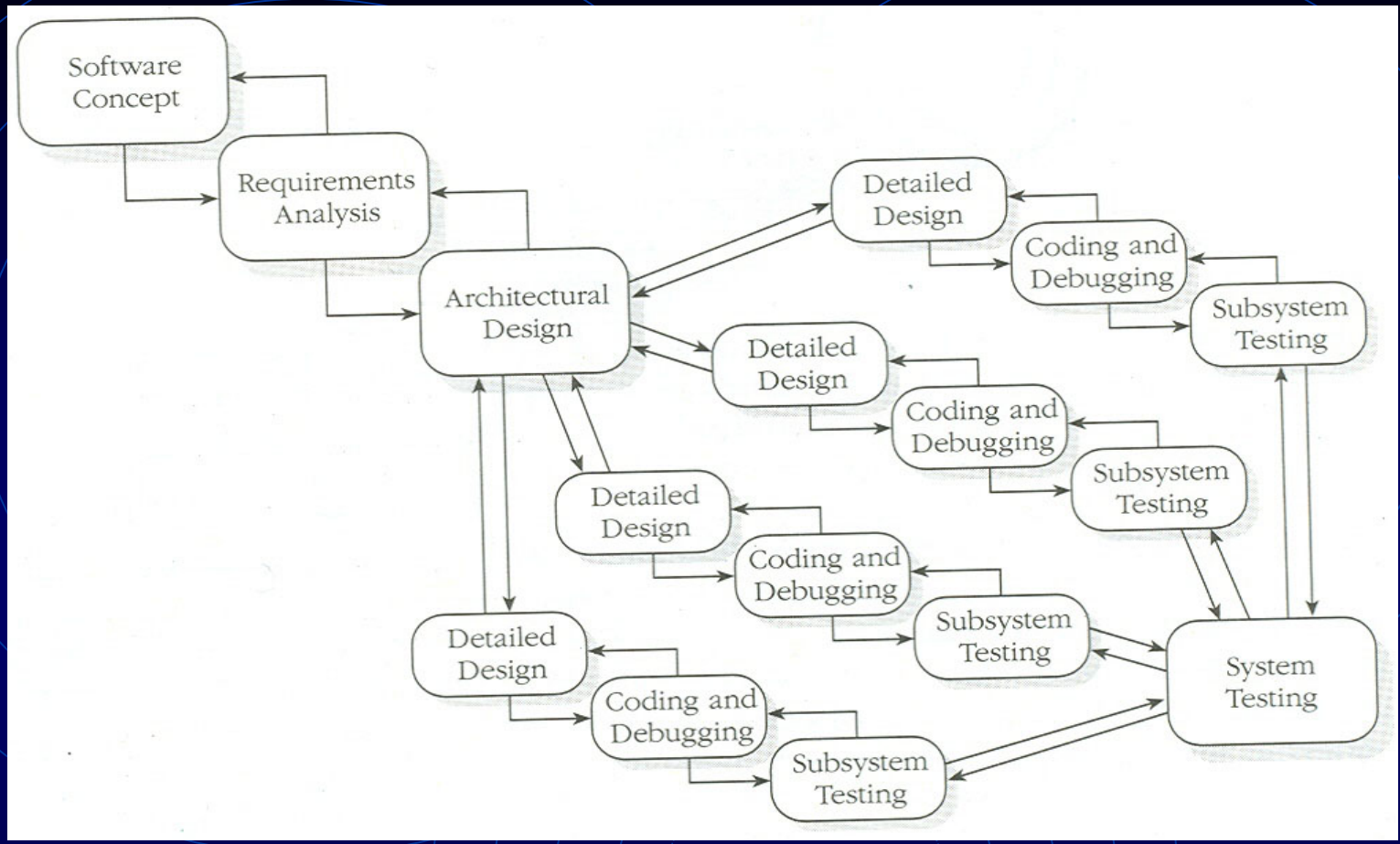

Waterfall Lifecycle Model

shamelessly reproduced without permission from

Steve McConnell, Rapid Development, ©1966



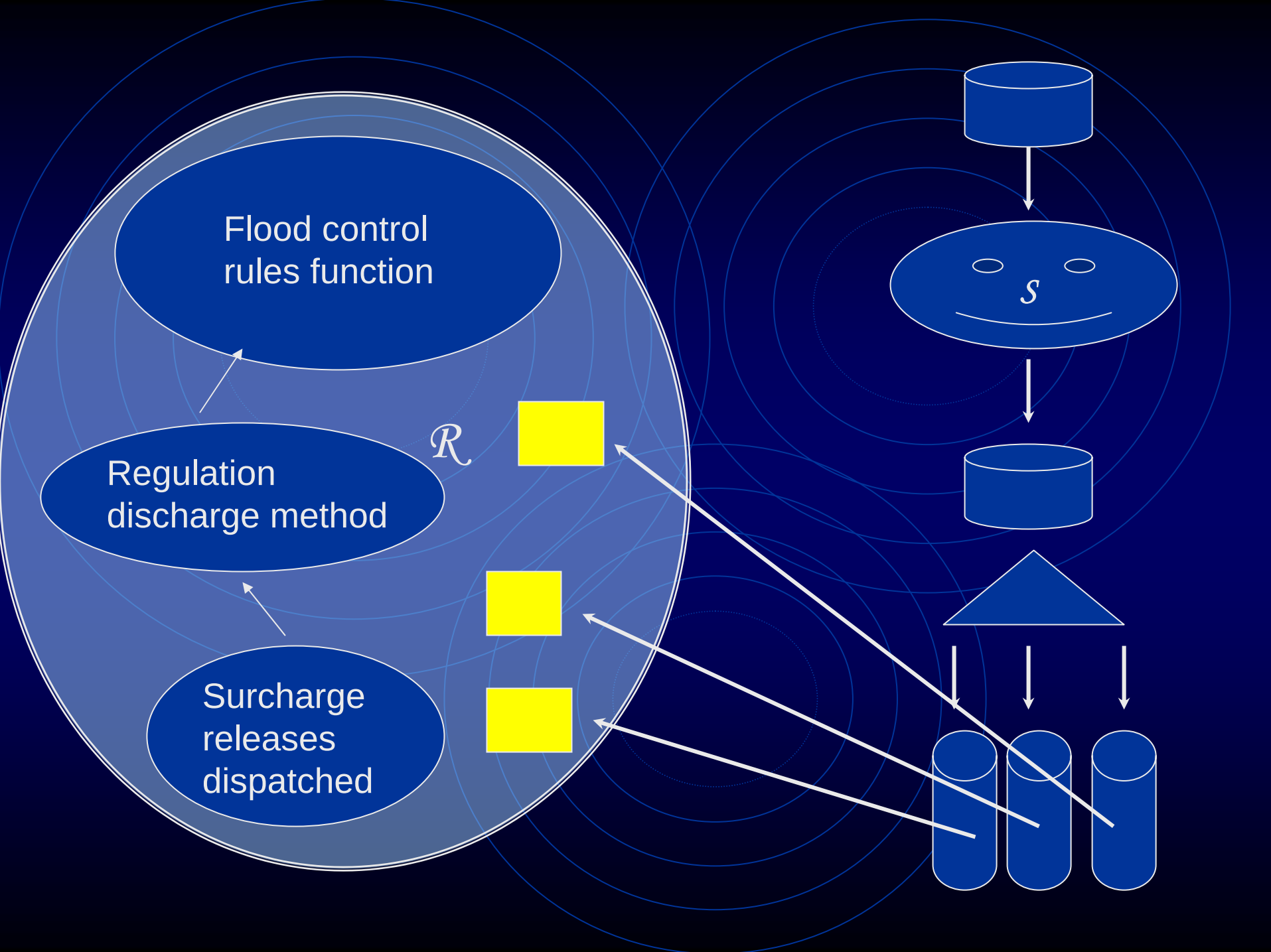
```
I = I + 1
IF(I.LE.LX) GO TO 9000
IF(J.EQ.12) GO TO 9001
J = J + 1
CALL SUB(J)
GOTO 80
```

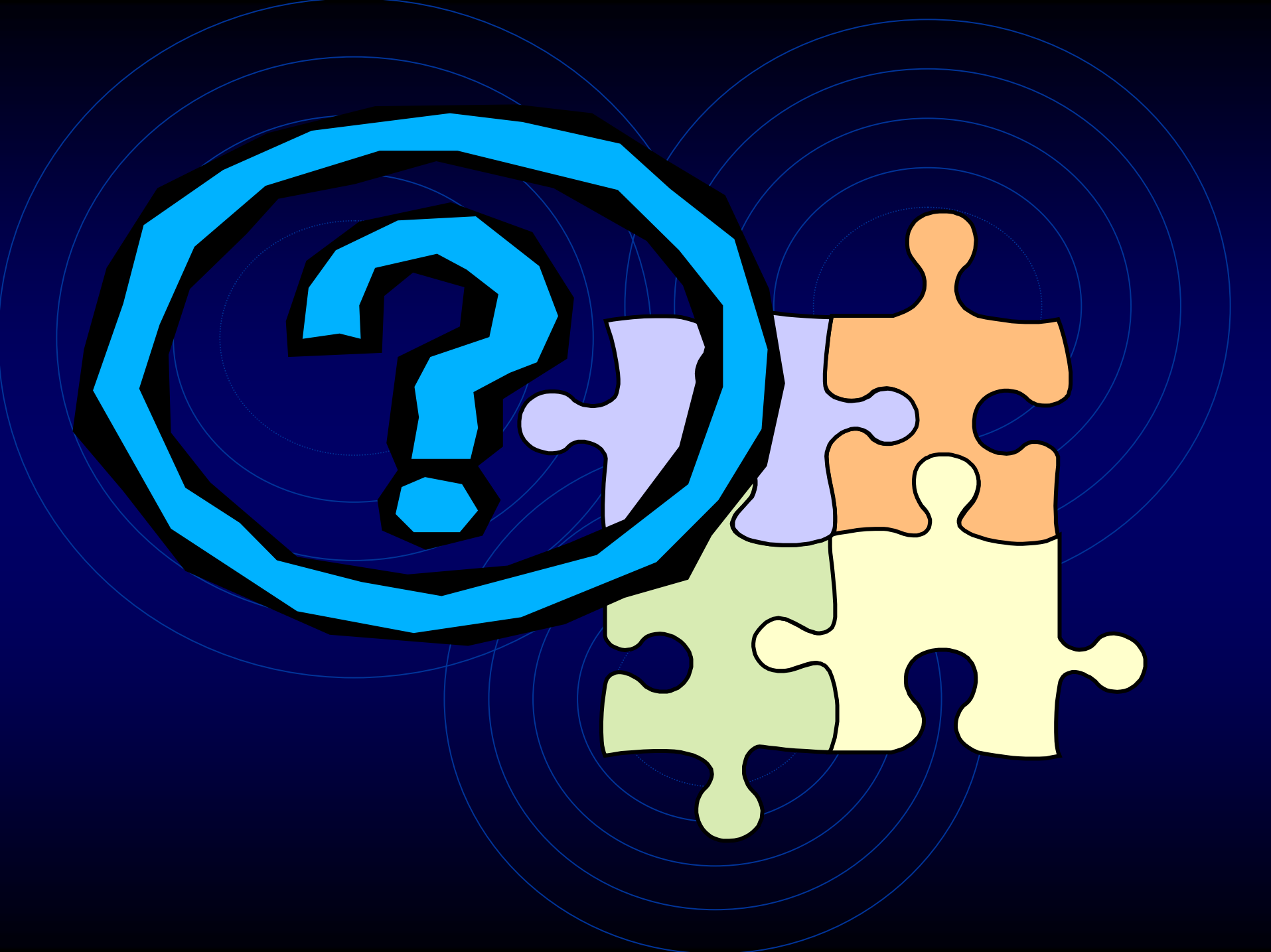



Waterfall-with-Projects Lifecycle Model

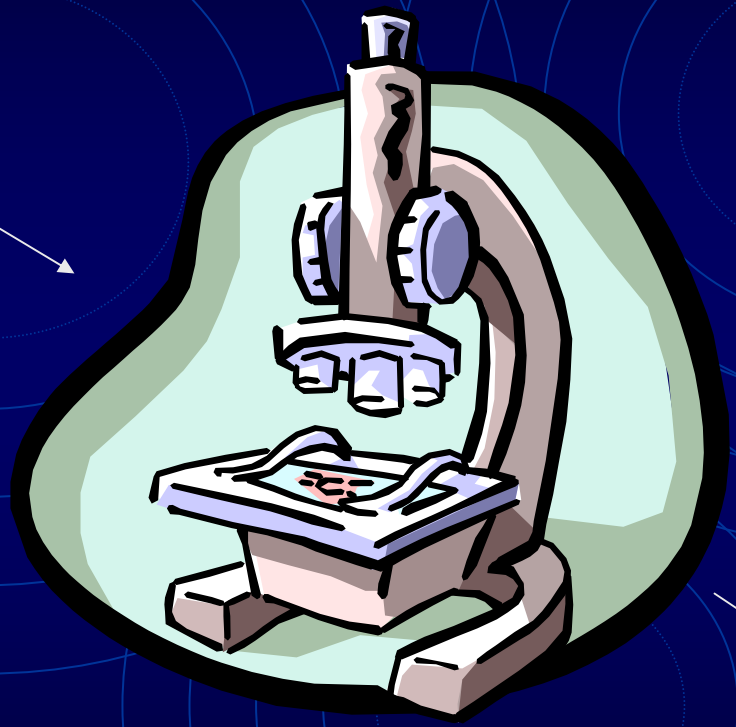
Shamelessly reproduced from

Steve McConnell, Rapid Development, © 1966


Flood control
rules function
$\mathcal{R}$
Regulation
discharge method
Surcharge
releases
dispatched
$S$



GOTO

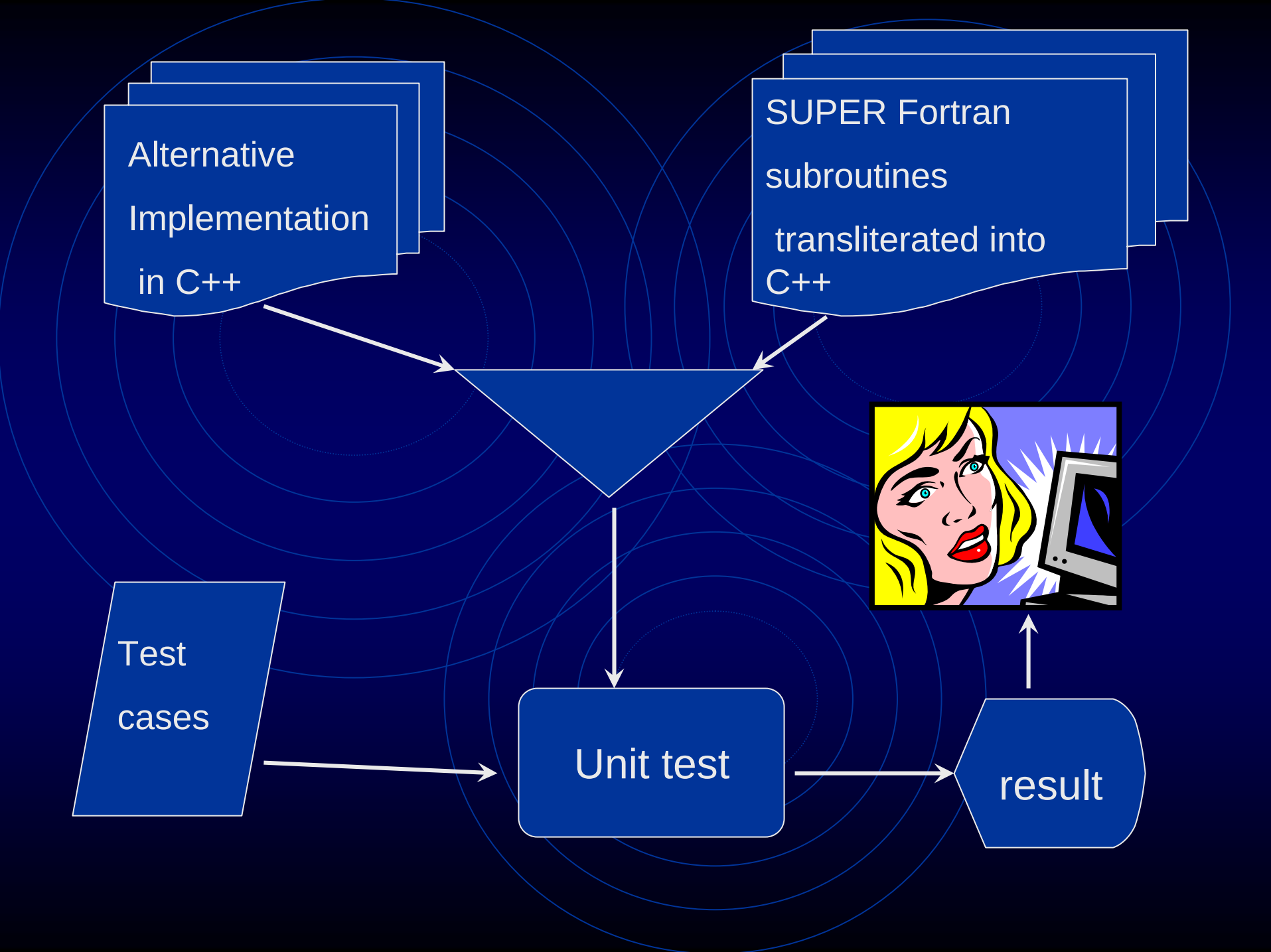


IF-THEN-ELSE

What saves a man is a step.

Then another step.

- Antoine de St. Exupery


Alternative
Implementation
in C++
SUPER Fortran
subroutines
transliterated into
C++
Test
cases
Unit test
result

Flood control
main algorithm

Stubbed-out objects,
methods,…
Parser.cpp, …

1-day
test
cases

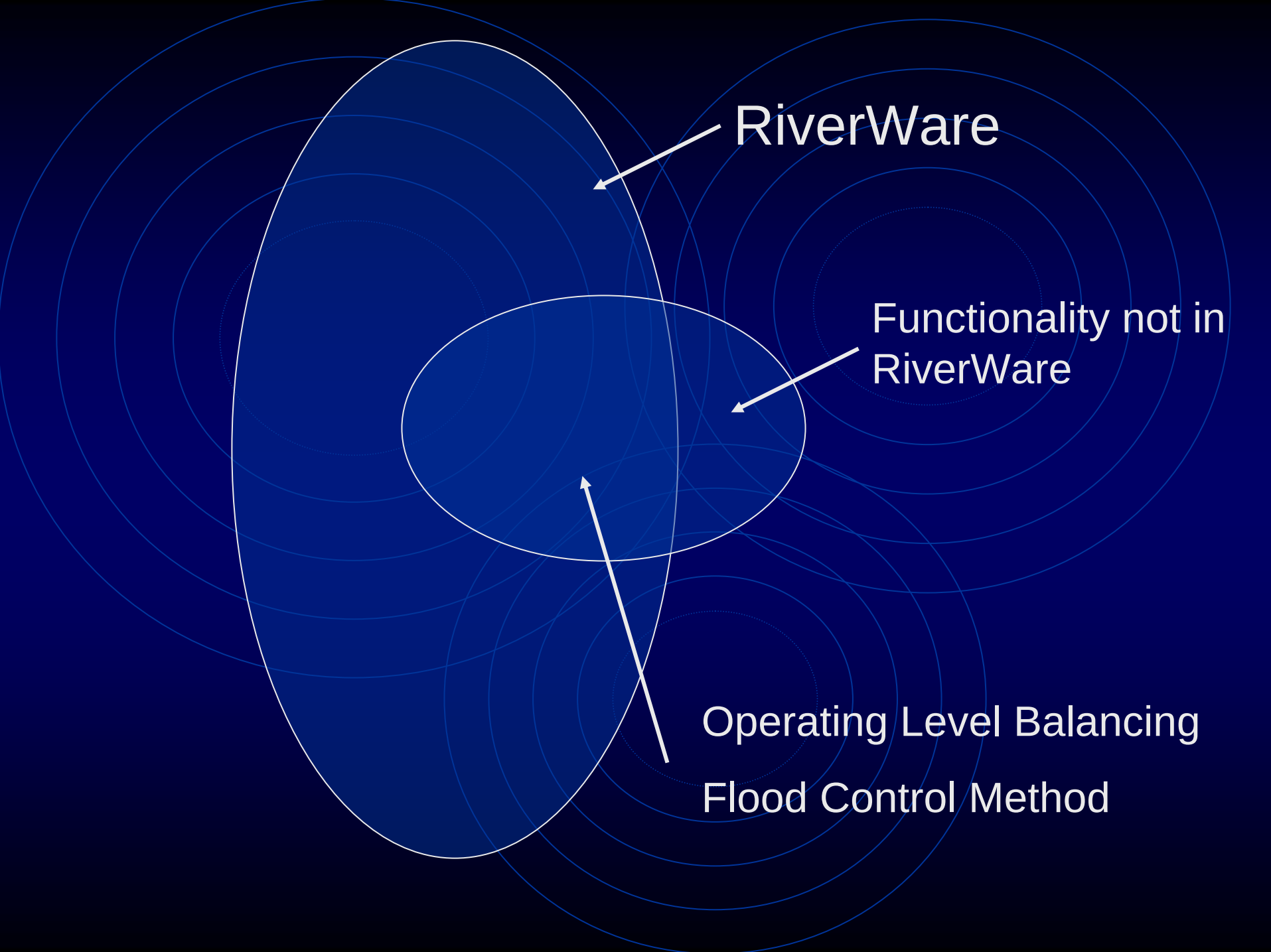
Unit test

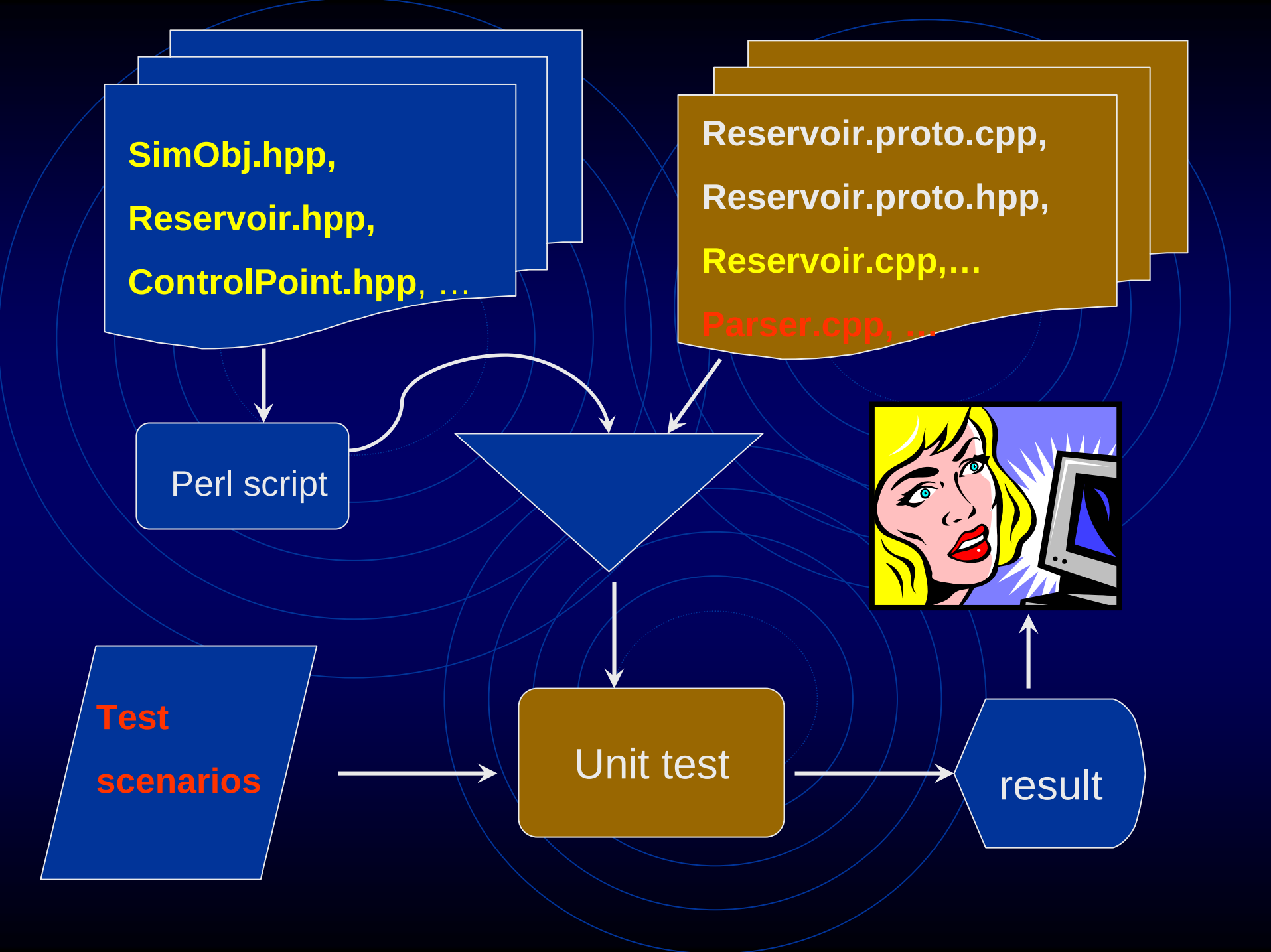
result

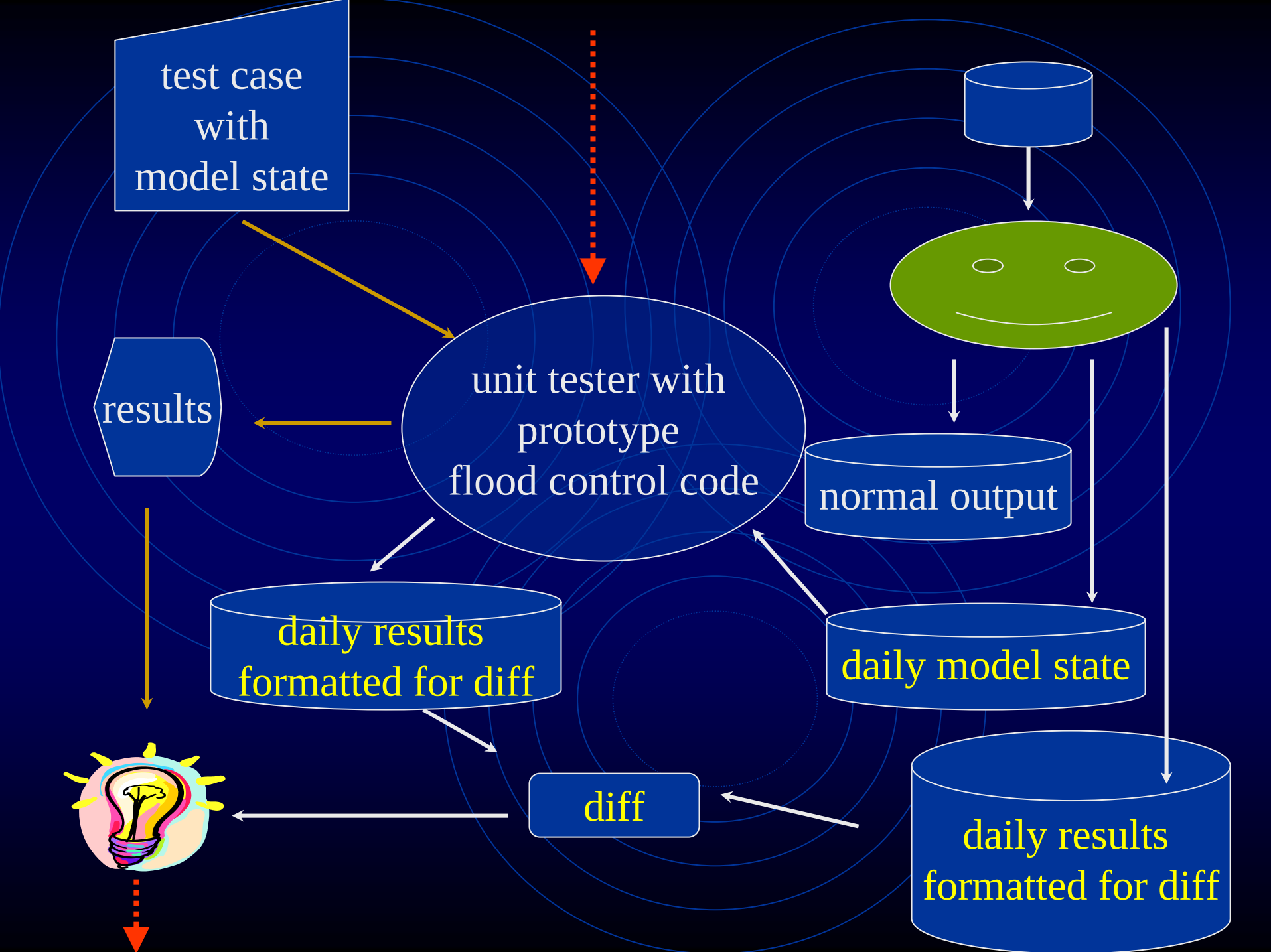
Proposed Design

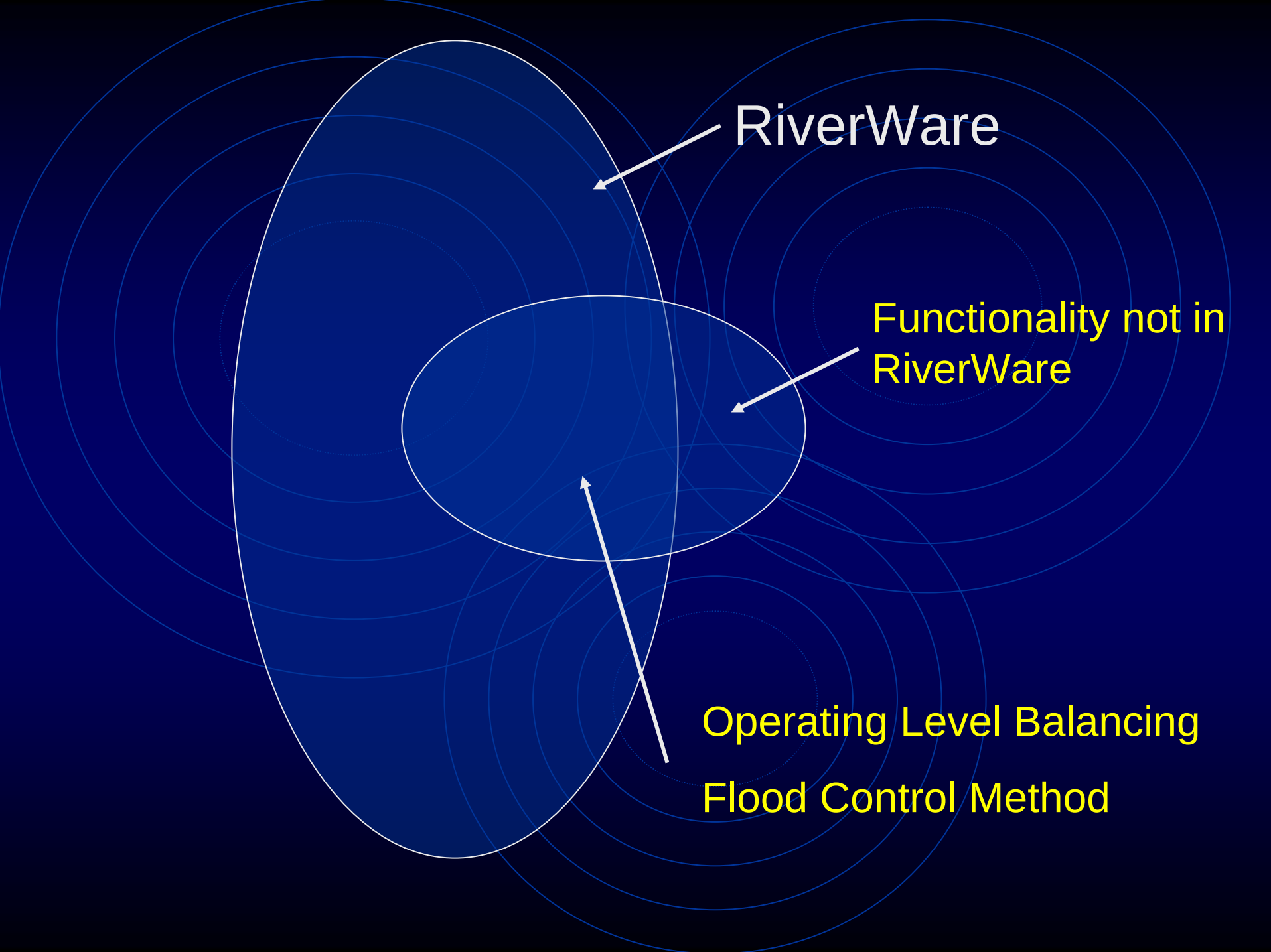
Ease of testing

Ease of developing unit tests


RiverWare
Functionality not in RiverWare
Operating Level Balancing
Flood Control Method


SimObj.hpp,
Reservoir.hpp,
ControlPoint.hpp, …
Reservoir.proto.cpp,
Reservoir.proto.hpp,
Reservoir.cpp,…
Parser.cpp, …
Perl script
Test
scenarios
Unit test
result


test case
with
model state
unit tester with
prototype
flood control code
results
normal output
daily results
formatted for diff
daily model state
diff
daily results
formatted for diff


RiverWare
Functionality not in RiverWare
Operating Level Balancing
Flood Control Method

If you’re not confused by all this,

it just shows you’re not thinking clearly.

-Anon