iMpLemENtiNg eVoLuTiONaRy ARcHitECtuREs

Day 2:
The Evolution of Penultimate Widgets!

@neal4d
nealford.com

@rebeccaparsons

@patkua

Evolutionary Architecture

An evolutionary architecture supports
guided,
incremental change
across multiple dimensions.

guided

***architectural** fitness function*:

An architectural fitness function provides an objective integrity assessment of some architectural characteristic(s).

Fitness Functions

Exercise:

What *type* of fitness function?

evolutionary architecture

governance

process

enterprise architecture

Agenda Part 2

The Evolution of Penultima⬆⊠e

Agenda Part 2

fitness functions

architecture migration

The Evolution of

Penultima⬆⊠e

experimentation

automating governance

hypothesis/data driven development

Penultima⬆⊠e

Penultima⬆⊠e

pe·nul·ti·mate

/pəˈnəltəmət/

adjective

last but one in a series of things; second to the last.
"the penultimate chapter of the book"
synonyms: next-to-last, second-to-last, second-last
"the penultimate movie on my top-ten list is an animated feature"

Penultima⬆⊠e

—many outdated systems

Penultima⬆⊠e

—many outdated systems
—outsourced operations

Penultima⬆⊠e

—many outdated systems
—outsourced operations
—impending competitors

Penultima⬆⊠e

—many outdated systems
—outsourced operations
—impending competitors
—second to last in market!

Penultima⬆⊠e

The Addison-Wesley Signature Series
A Martin Fowler Signature Book
Continuous Delivery
Reliable Software Releases through Build, Test, and Deployment Automation
Jez Humble
David Farley
Foreword by Martin Fowler

The Addison-Wesley Signature Series
A Martin Fowler Signature Book
Continuous Delivery
Reliable Software Releases through Build, Test, and Deployment Automation
Jez Humble
David Farley
Foreword by Martin Fowler

The Addison-Wesley Signature Series
A Martin Fowler Signature Book
Refactoring Databases
Evolutionary Database Design
Scott W. Ambler
Pramod J. Sadalage
Forewords by Martin Fowler, John Graham, Sachin Rekhi, and Dr. Paul Dorsey

The Addison-Wesley Signature Series
A Martin Fowler Signature Book
Continuous Delivery
Reliable Software Releases through Build, Test, and Deployment Automation
Jez Humble
David Farley
Foreword by Martin Fow

THE LEAN SERIES
ERIC RIES, SERIES EDITOR
Jez Humble, Joanne Molesky & Barry O'Reilly
LEAN ENTERPRISE
How High Performance Organizations Innovate at Scale
O'REILLY

The Addison-Wesley Signature Series
A Martin Fowler Signature Book
Refactoring Databases
Evolutionary Database Design
Scott W. Ambler
Pramod J. Sadalage
Forewords by Martin Fowler, John Graham, Sachin Rekhi, and Dr. Paul Dorsey

The Addison-Wesley Signature Series
A Martin Fowler Signature Book
Continuous Delivery
Reliable Software Releases through Build, Test, and Deployment Automation
Jez Humble
David Farley
Foreword by Martin Fow

THE LEAN SERIES
ERIC RIES, SERIES EDITOR
Jez Humble, Joanne Molesky & Barry O'Reilly
LEAN ENTERPRISE
How High Performance Organizations Innovate at Scale
O'REILLY

The Addison-Wesley Signature Series
A Martin Fowler Signature Book
Refactoring Databases
Evolutionary Database Design
Scott W. Ambler
Pramod J. Sadalage
Forewords by Martin Fowler, John Graham, Sachin Rekhi, and Dr. Paul Dorsey

The Pragmatic Programmers
Release It!
Second Edition
Design and Deploy Production-Ready Software
Michael T. Nygard
Edited by Katharine Dvorak

The Addison-Wesley Signature Series
A Martin Fowler Signature Book
Continuous Delivery
Reliable Software Releases through Build, Test, and Deployment Automation
Jez Humble
David Farley
Foreword by Martin Fow

THE LEAN SERIES
ERIC RIES, SERIES EDITOR
Jez Humble, Joanne Molesky & Barry O'Reilly
LEAN ENTERPRISE
How High Performance Organizations Innovate at Scale
O'REILLY

The Addison-Wesley Signature Series
A Martin Fowler Signature Book
Refactoring Databases
Evolutionary Database Design
Scott W. Ambler
Pramod J. Sadalage
Forewords by Martin Fowler, John Graham, Sachin Rekhi, and Dr. Paul Dorsey

The Pragmatic Programmers
Release It!
Second Edition
Design and Deploy Production-Ready Software
Michael T. Nygard
Edited by Katharine Dvorak

O'REILLY
Infrastructure as Code
MANAGING SERVERS IN THE CLOUD
Kief Morris

The Addison-Wesley Signature Series
Continuous Delivery
Reliable Software Releases through Build, Test, and Deployment Automation
Jez Humble
David Farley
Foreword by Martin Fowler

O'REILLY
Building Evolutionary Architectures
SUPPORT CONSTANT CHANGE
Neal Ford, Rebecca Parsons & Patrick Kua

FACTORING
TABASES
UTIONARY
BASE DESIGN
V. AMBLER
J. SADALAGE
Forewords by Martin Fowler, John Graham, Sachin Rekhi, and Dr. Paul Dorsey

The Pragmatic Programmers
Release It!
Second Edition
Design and Deploy Production-Ready Software
Michael T. Nygard
Edited by Katharine Dvorak

THE LEAN SERIES
ERIC RIES, SERIES EDITOR
Jez Humble, Joanne Molesky & Barry O'Reilly
LEAN ENTERPRISE
How High Performance Organizations Innovate at Scale
O'REILLY

O'REILLY
Infrastructure as Code
MANAGING SERVERS IN THE CLOUD
Kief Morris

Penultima⬆⊠e

evolutionary architecture

automated
machine provisioning

Penultima⬆⊠e

Penultima⬆⊠e

too complex!!

Penultima⬆⊠e

too complex!!

no expertise!!

Penultima⬆⊠e

too complex!!

no expertise!!

where to start?!!

Penultima⬆⊠e

Penultima⬆⊠e

too complex!!

no expertise!!

where to start?!!

(it'll never work)

too complex!!

no expertise!!

where to start?!!

(it'll never work)

Penultima⬆⊠e

demonstration defeats discussion

Penultima⬆⊠e

microservices for
new development

migrate critical
existing systems

component coupling

the extent to which components know about each other

Structured Design
Fundamentals of a Discipline of Computer Program and Systems Design
Edward Yourdon / Larry L. Constantine
YOURDON PRESS COMPUTING SERIES

component coupling

component coupling

pathological coupling

one component relies on the inner workings of another component

component coupling

external coupling

multiple components share an externally imposed protocol or data format

component coupling

control coupling

one component passes information to another component on what to do

component coupling

component coupling

afferent coupling

the degree to which other components are dependent on the target component

component coupling

efferent coupling

the degree to which the target component
is dependent on other components

Abstractness

$$A = \frac{\sum m^a}{\sum m^c}$$

where:
m^a => abstract elements
m^c => concrete elements

Instability

$$I = \frac{C^e}{C^e + C^a}$$

where:
c^e => efferent coupling
c^a => afferent coupling

Distance from the Main Sequence

$$D = |A + I - 1|$$

Distance from the Main Sequence

$$D = |A + I - 1|$$

Distance from the Main Sequence

$$D = |A + I - 1|$$

distance from the main sequence

Dependency Metrics

Connascence

https://connascence.io/name.html

connascence.io

- Static Connascences
- Name
- Type
- Meaning
- Position
- Algorithm
- Dynamic Connascences
- Execution
- Timing
- Value
- Identity
- Properties
- Strength
- Locality
- Degree
- Other Resources

What is Connascence?

Connascence is a software quality metric & a taxonomy for different types of coupling. This site is a handy reference to the various types of connascence, with examples to help you improve your code.

Subject to Change

All code is subject to change. As the real world changes, so too must our code. Connascence gives us an insight into the long-term impact our code will have on flexibility, as we write it. Maintaining a flexible codebase is essential for maintaining long-term development velocity.

A Flexible Metric

Connascence is a metric, and like all metrics is an imperfect measure. However, connascence takes a more holistic approach, where each instance of connascence in a codebase must be considered on three separate axes:

1. Strength. Stronger connascences are harder to discover, or harder to refactor.
2. Degree. An entity that is connascent with thousands of other entities is likely to be a larger issue than one that is connascent with only a few.
3. Locality. Connascent elements that are close together in a codebase are better than ones that are far apart.

The three properties of Strength, Degree, and Locality give the programmer all the tools they need in order to make informed decisions about when they will permit certain types of coupling, and when the code ought to be refactored.

A Vocabulary for Coupling

Arguably one of the most important benefits of connascence is that it gives developers a vocabulary to talk about different types of coupling. Connascence codifies what many experienced engineers have learned by trial and error: Having a common set of nouns to refer to different types of coupling allows us to share that experience more easily.

This work is licensed under a Creative Commons Attribution-ShareAlike 3.0 Unported License.

Origin

Two components are connascent if a change in one would require the other to be modified in order to maintain the overall correctness of the system.

static

dynamic

static

Connascence of Name

multiple components must agree on the name of an entity

```
class Request:

    def __init__(self, url, data=None, headers={},
                 origin_req_host=None, unverifiable=False,
                 method=None):
        pass

    def set_proxy(self, host, type):
        pass
```

static

Connascence of Type

multiple components must agree on the type of an entity

```
std::string cost;

cost = 10.95; // OOPS!
```

```
calculate_age(1, 9, 1984)
calculate_age(1, 9, 84)
calculate_age('1', '9', '1984')
calculate_age('1', 'September', '1984')
```

```
def calculate_age(birth_day, birth_month, birth_year):
    # do the calculation here:
```

static

Connascence of Type

multiple components must agree on the type of an entity

selective typing via clojure-spec

https://clojure.org/guides/spec

```
(ns my.domain (:require [clojure.spec.alpha :as s]))
(def email-regex #"^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,63}$")
(s/def ::email-type (s/and string? #(re-matches email-regex %)))

(s/def ::acctid int?)
(s/def ::first-name string?)
(s/def ::last-name string?)
(s/def ::email ::email-type)

(s/def ::person (s/keys :req [::first-name ::last-name ::email]
                        :opt [::phone]))
```

static

Connascence of Meaning

multiple components must agree on the meaning of particular values

```
def is_credit_card_number_valid(card_number):
    # Check for 'test' credit card numbers:
    if card_number == "9999-9999-9999-9999":
        return True
    # Do normal validation:
    # ...
```

static

Connascence of Meaning

multiple components must agree on the meaning of particular values

```
def get_user_role(username):
    user = database.get_user_object_for_username(username)
    if user.is_admin:
        return 2
    elif user.is_manager:
        return 1
    else:
        return 0
```

```
if get_user_role(username) != 2:
    raise PermissionDenied("You must be an administrator")
```

constants!

…move from connascence of *meaning* to connascence of *name*

static

Connascence of Position

multiple entities must agree on the order of values

```
def get_user_details():
    # Returns a user's details as a list:
    # first_name, last_name, year_of_birth, is_admin
    return ["Thomas", "Richards", 1984, True]
```

```
placeOrder(name, address, amount, discount)
```

"Neal" "Seat 10C"

"Seat 10C" "Neal"

static

Connascence of Algorithm

multiple components must agree on a particular algorithm

```
def write_data_to_cache(data_string):
    with open('/path/to/cache', 'wb') as cache_file:
        cache_file.write(data_string.encode('utf8'))
```

```
def read_data_from_cache():
    with open('/path/to/cache', 'rb') as cache_file:
        return cache_file.read().decode('utf8')
```

Origin

Two components are connascent if a change in one would require the other to be modified in order to maintain the overall correctness of the system.

static

dynamic

dynamic

Connascence of Execution

the order of execution of multiple components is important

```
email = Email()
email.setRecipient("foo@example.comp")
email.setSender("me@mydomain.com")
email.send()
email.setSubject("Hello World")
```

dynamic

Connascence of Timing

the timing of the execution of multiple components is important

dynamic

Connascence of Value

occurs when several values relate on one another
and must change together

dynamic

Connascence of Value

occurs when several values relate on one another and must change together

dynamic

Connascence of Identity

multiple components must reference the same entity

dynamic

Connascence of Identity

multiple components must reference the same entity

Connascence

Two components are connascent if a change in one would require the other to be modified in order to maintain the overall correctness of the system.

static

dynamic

Contranascence

connascence of difference

negative connascence

difference rather than equality must be preserved.

```
int i;
int j;
```

Contranascence

difference rather than equality must be preserved.

Connascence Properties

static

dynamic

Strength

Locality

Degree

Connascence Properties

Strength

name

type

meaning

algorithm

position

refactor
this way

execution order

timing

value

identity

static

dynamic

Connascence Properties

static

dynamic

Strength

Locality

Degree

Connascence Properties

Locality

static

Proximal code (in the same module, class, or function) should have more & higher forms of connascence than less proximal code (in separate modules, or even codebases).

dynamic

Coupling

semantic

syntactic

Connascence Properties

Locality

locality matters!

Coupling

syntactic

Connascence Properties

static

dynamic

Strength

Locality

Degree

Connascence Properties

Degree

Connascence Properties

static

dynamic

Strength

Locality

Degree

Component Cohesion

the degree and manner to which the operations of a component are related to one another

Chidamber & Kemerer Metrics

✓ LCOM (Lack of Cohesion in Methods)

$$LCOM96b = \frac{1}{a}\sum_{j=1}^{a} \frac{m - \mu(A_j)}{m}$$

Chidamber & Kemerer Metrics

✓ LCOM (Lack of Cohesion in Methods)

The LCOM is a count of the number of method pairs whose similarity is 0 (i.e. σ() is a null set) minus the count of method pairs whose similarity is not zero.

Chidamber & Kemerer Metrics

✓ LCOM (Lack of Cohesion in Methods)

The LCOM is a count of the number of method pairs whose similarity is 0 (i.e. σ() is a null set) minus the count of method pairs whose similarity is not zero.

The sum of sets of methods not shared via sharing fields.

Chidamber & Kemerer Metrics

✓LCOM (Lack of Cohesion in Methods)

Chidamber & Kemerer Metrics

✓LCOM (Lack of Cohesion in Methods)

Chidamber & Kemerer Metrics

✓LCOM (Lack of Cohesion in Methods)

Chidamber & Kemerer Metrics

✓LCOM (Lack of Cohesion in Methods)

Unifying Coupling/Connascence

When Migrating Architecture…

1. improve modularity

When Migrating Architecture…

1. improve modularity

2. create service-based architecture

When Migrating Architecture…

1. improve modularity

2. create service-based architecture

3. isolate quanta

When Migrating Architecture…

1. improve modularity

2. create service-based architecture

3. isolate quanta

4. evolve via microservices

Migration Challenges

service component granularity

Migration Challenges

service component granularity

Migration Challenges

service component granularity

coarse-grained service components address transactional issues but may not achieve your desired goals

Migration Challenges

service component granularity

Migration Challenges

service component granularity

Migration Challenges

service component granularity

fine-grained service components may lead to too much orchestration and inter-dependency between service components

Migration Challenges

service component granularity

Migration Challenges

service component granularity

business functionality groupings

Migration Challenges

service component granularity

business functionality groupings

transactional boundaries

Migration Challenges

service component granularity

business functionality groupings

transactional boundaries

deployment goals

Migration Challenges

service component granularity

business functionality groupings

transactional boundaries

deployment goals

scalability needs

Migration Challenges

shared components and object hierarchies

Migration Challenges

shared components and object hierarchies

Electronics Recycling Application

Electronics Recycling Application

Electronics Recycling Application

Migration Challenges

shared components and object hierarchies

developers can *split* modules....

Migration Challenges

shared components and object hierarchies

...or move into a shared library or jar file...

Migration Challenges

shared components and object hierarchies

…or replicate in each service component

Migration Challenges

shared components and object hierarchies

Migration Challenges

shared component techniques

jar /dll
(compile or runtime)

shared library
(compile time)

code replication

remote services

Migration Targets

Microservices?

Service-based Architecture

Service-based Architecture

service granularity

database scope

Electronics Recycling Application

Electronics Recycling Application

Electronics Recycling Application

Service-based Architecture

adding microservices

Service-based Architecture

adding microservices

Shared-database Integration

Refactoring Databases

Move Column Refactoring

Original Schema

Transition Period

Transition Period

Ending Schema

Resulting Schema

Decouple DB Updates: the Expand/contract Pattern

Expand/Contract Pattern

Change ‘name’ to ‘firstname’ & ’lastname’

Change ‘name’ to ‘firstname’ & ’lastname’

Change ‘name’ to ‘firstname’ & ’lastname’

#1: ~~integration points~~, ~~legacy data~~

```
ALTER TABLE customer ADD firstname VARCHAR2(60);
ALTER TABLE customer ADD lastname VARCHAR2(60);
ALTER TABLE customer DROP COLUMN name;
```

#2: ~~integration points~~, legacy data

```
ALTER TABLE Customer ADD firstname VARCHAR2(60);
ALTER TABLE Customer ADD lastname VARCHAR2(60);
UPDATE Customer set firstname = extractfirstname (name);
UPDATE Customer set lastname = extractlastname (name);
ALTER TABLE customer DROP COLUMN name;
```

#3: integration points, legacy data

```
ALTER TABLE Customer ADD firstname VARCHAR2(60);
ALTER TABLE Customer ADD lastname VARCHAR2(60);

UPDATE Customer set firstname = extractfirstname (name);
UPDATE Customer set lastname = extractlastname (name);

CREATE OR REPLACE TRIGGER SynchronizeName
BEFORE INSERT OR UPDATE
ON Customer
REFERENCING OLD AS OLD NEW AS NEW
FOR EACH ROW
BEGIN

IF :NEW.Name IS NULL THEN
  :NEW.Name := :NEW.firstname||' '||:NEW.lastname;
END IF;
IF :NEW.name IS NOT NULL THEN
  :NEW.firstname := extractfirstname(:NEW.name);
  :NEW.lastname := extractlastname(:NEW.name);
END IF;
END;
```

Migration Targets

1. Top Level Partitioning?

domain

modular monolith

http://www.codingthearchitecture.com/presentations/sa2015-modular-monoliths

technical

layered architecture

2. Natural Seams

- Domains
- Transactions
- Metrics

Finding Seams

Tactical Forking

https://www.thoughtworks.com/profiles/fausto-de-la-torre

Tactical Forking

initial state

Tactical Forking

Tactical Forking

initial state

first step benefit is tangible

constant refactor

Tactical Forking

Penultima⬆⊠e

microservices for new development

migrate critical existing systems

Penultima⬆⊠e

Deployment Pipeline

cycle time
01001001010101
01010101010101
00101010010010
00100100010001
replicating CI
containerize & deploy
database
atomic fitness functions
database
holistic fitness functions
database
security review
audit
deploy to production

cycle time
lead time

Penultima⬆⊠e

fitness function:

CycleTimeGuard

PenultimateWidgets has entered an extremely competitive market with a startup running popular flash promotions. The marketing department committed to keeping up with PenultimateWidgets own aggressive marketing campaign, which requires development teams to be able to deliver changes to the promotion pages within an hour. Thus, they want a guarantee that changes can always take place within an hour.

Penultima⬆⊠e

fitness function:

CycleTimeGuard

- utilize the good modularity of the monolith to build a deployment pipeline that automatically incorporates CMS changes
- continuously deploy new content for marketing pages as long as they pass smoke test
- measure cycle time from deployment pipeline for deployments for marketing sites
- save cycle time results to aggregate a weekly chart

Incidentally Coupled Teams

user interface

server-side

DBA

Conway's Law

“organizations which design systems ... are constrained to produce designs which are copies of the communication structures of these organizations”

Melvin Conway, 1968

en.wikipedia.org/wiki/Conway%27s_law

Low Efferent Coupling between Teams

presentation
component
component
component
business rules
component
component
component
persistence
component
component
component
database

API/Facade
module
module
module
CatalogCheckout
module
module
module
database
OrderFulfillmert
module
module
module
database

API/Facade
module
module
module
CatalogCheckout
module
module
module
database
OrderFulfillmert
module
module
module
database

API/Facade
module
module
module
CatalogCheckout
module
module
module
database
OrderFulfillmert
module
module
module
database

Inverse Conway Maneuver

Experimentation is the only way to discover if something works within a particular organization.

Retrofitting Governance

Penultima⬆☒e

Penultima⬆⊠e

fitness function:

Configure Some of the Things All of the Time

PenultimateWidgets has been burned in the past by having hard-coded constants in several places in their code base, governing both business cases (thresholds, maximums, etc.) and operational concerns. Architects need a way to prevent developers from accidentally hard-coding critical values.

Penultima⬆⊠e

fitness function:

Configure Some of the Things All of the Time

- Upon check-in:
  - scan the list of dynamic values in the code base and see if any of them are assigned to (or their setter is called) or a new instance is instantiated via a constructor that provides values
  - ensure that all assignments for dynamic values are assigned via the API that handles configuration, not with hard-coded values
  - scan the definitions of infrastructure to look for dynamic values and ensure they are assigned properly before building the environment
  - scan the configuration file and ensure that all defined dynamic values are present and have values

Penultima⬆⊠e

fitness function:

Audit the Accounting

PenultimateWidgets has been a Bad Company (cheating on some complex accounting rules), but they have reformed their ways and paid their penalties. Now, however, as part of the settlement, they agree to put auditing measures in place around code that handles accounting rules. State auditors come once a quarter to look everything over, but PenultimateWidgets has found it too much of a chore to delay preparation until near audit time; they want a more automated way to verify things.

Penultima⬆⊠e

fitness function:

Audit the Accounting

- During check-in, look at git logs to see if any files in the affected packages were modified
- for any modified file, add it to a database of files to check by the auditors
- the tech lead code reviews changes for the last week before standup
- at audit time, show each of the changed files to auditors and allow them access to the repository to verify that no other files were changed

Penultima⬆⊠e

fitness function:

Debug All the Things

and Javascript applications support remote debugging in the development envir

Penultima⬆⊠e

fitness function:

Debug All the Things

- for Java applications, create a monitor that runs in development and QA that checks to ensure the remote debugging ports are available and listening for all deployed Java applications
- create a symmetrical monitor for production systems to ensure the remote debugging ports are NOT enabled in production
- For Javascript, create a special flag variable declaration at the top of the file that is not used anywhere else (meaning the minimizer will remove it). For development and QA environments, check to make sure the special flag variable is present
- include source maps for obfuscated code to map problems back to the source

What to Port?

File Actions Options Help
Name New Name Sub... Type Size Created Modifi... Acce... Length
Microsoft Analysis Services
Microsoft Games
Microsoft Help Viewer
Microsoft IntelliType Pro
Microsoft Office
Microsoft SQL Server
Microsoft SQL Server Com...
Microsoft Sync Framework
RegEx (1)
Repl. (3)
Remove (5)
Add (7)
Auto Date (8)
Numbering (10)
File (2)
Case (4)
Move/Copy (6)
Append Folder Name (9)
Extension (11)
Selections (12)
New Location (13)
Reset
Revert
Rename
Love Bulk Rename Utility? Try ViceVersa PRO, file sync and backup software for Windows. Click Here To Find Out More...
43 Objects (1 Selected)
Prefix filenames with

Cash Flow
Reports
Companies
Settings
Users
Account
My Info
Logout
Help
Cash Flow
Add Income
Add Expense
Export
Month
Week
List
Apr 09 - Nov 09
April
May
June
July
August
September
October
November
CASH ON HAND
INCOME
CONSULTING
Microsoft
The Wall Street Journal
Interface Design & Development
PRODUCTS
Merchandise
Web Applications
SERVICES
Add income
EXPENSES
Foxnews.com
Microsoft
Brand Consulting
Web Applications
INTERNET/TELECOMMUNICATIONS
PAYROLL
RENT
Office Space
Add expense

Experimentation!

- add logging to the existing application to aggregate what menu items users choose and the user group
- publish the log to a known URL at PenultimateWidgets

- aggregate the logs to get usage and user coverage

Penultima⬆⊠e

fitness function:

Make Sure New Really Replaces Old

PenultimateWidgets legacy accounting system grew over many years, with arcane business rules scattered in many places—no one really understands how everything works anymore. They worked for several years replacing it, but worry that the new system cannot replicate the byzantine business rules that their clients have come to depend upon.

Penultima⬆⊠e

fitness function:

Make Sure New Really Replaces Old

- For each discrete function covered under this rule, create a functional test that treats each system (old and new) as a black box, looking only at the results of each.
- Provision the old system (perhaps using the Legacy in the Box pattern) and new in an integration testing environment and run the functional replacement tests.
- Stories are not marked complete until the functional tests run.

Penultima⬆☒e

fitness function:

Replace the Crufty Core

PenultimateWidgets built a network scanning tool many, many years ago to look for suspicious activity at the network packet level. To achieve the proper performance, the code was written and has been maintained in increasingly complex C code. PenultimateWidgets needs to replace the code but is nervous about introducing bugs and/or harming performance or throughput.

Penultima⬆⊠e

fitness function:

Replace the Crufty Core

- Built fitness functions around the existing C code to ensure performance and throughput values.
- Build a spike solution in both Java and Go (and perhaps others) and run them against the fitness functions to see if they can meet the threshold
- Use Scientist (https://github.com/github/scientist) port Scientist4J (https://github.com/rawls238/Scientist4J) to verify the new code against the old code in production

While We're at it…

Penultima⬆⊠e

fitness function:

A11y All the Things

PenultimateWidgets has decided that all new development of user interfaces must support accessibility, both web and desktop.

Penultima⬆⊠e

fitness function:

A11y All the Things

- add a fitness function to the deployment pipeline for web projects that runs pa11y (http://pa11y.org/) to ensure compliance
- for Java, use PMD to check code within the user interface and look for control instantiation
- check to make sure every control has the AccessibleName property set

Mitigating the Unknown

reuse

Software reuse is more like an organ transplant than snapping together Lego blocks.

John D. Cook

http://www.johndcook.com/blog/2011/02/03/lego-blocks-and-organ-transplants/

Monolithic user interface
business services
BS
BS
BS
BS
BS
message bus
process choreographer
service orchestrator
enterprise services
BS
BS
BS
BS
BS
application services
AS
infrastructure services
IS

auto and homeowners insurance division
commercial insurance division
casualty insurance division
customer service
life insurance division
disability insurance division
travel insurance division

auto and homeowners insurance division
customer
commercial insurance division
customer
casualty insurance division
customer
customer service
life insurance division
customer
disability insurance division
customer
travel insurance division
customer

Code Reuse (Over Time)

eusable code is, the less

API Layer
checkout
module
module
database
ship
module
module
database
inventory
module
module
database
listing
module
module
database
accounts
module
module
database
bulk txn
module
module
database
service
module
module
database
routing
module
module
database

hexagonal pattern

operational reuse

the sidecar pattern

the sidecar pattern

service mesh

two reuse solutions

SOA

service mesh

Data Mesh

https://martinfowler.com/articles/data-monolith-to-mesh.html

data warehouse

Data Lake

data lake on the cloud

Data Mesh

DATA MESH ARCHITECTURE

Data Mesh

https://martinfowler.com/articles/data-monolith-to-mesh.html

Penultima⬆⊠e

fitness function:

Elasticity of Microservices

PenultimateWidgets utilizes a microservices architecture and want to ensure that as new requests appear that the architecture elastically scales.

Penultima⬆⊠e

fitness function:

Elasticity of Microservices

microservices architecture and want to ensure that as new requests appear that th

Penultima⬆⊠e

fitness function:

Elasticity of Microservices

microservices architecture and want to ensure that as new requests appear that th

Penultima⬆⊠e

fitness function:

Elasticity of Microservices

microservices architecture and want to ensure that as new requests appear that th

Penultima⬆⊠e

fitness function:

Elasticity of Microservices

microservices architecture and want to ensure that as new requests appear that th

Penultima⬆⊠e

fitness function:

Elasticity of Microservices

microservices architecture and want to ensure that as new requests appear that th

Penultima⬆⊠e

fitness function:

Elasticity of Microservices

microservices architecture and want to ensure that as new requests appear that th

Penultima⬆⊠e

fitness function:

Elasticity of Microservices

- build a Hello, World service to allow instantiation of the service template
- deploy the service and direct enough requests to it to trigger elasticity in the naming service
- verify that the number of instances rose properly

Penultima⬆⊠e

fitness function:

Deterministic Distributability

coded strings in applications that tie applications to a particular data center locati

Penultima⬆⊠e

fitness function:

Deterministic Distributability

- Require that developers use configuration files for all database configuration (using tools like ArchUnit)
- During testing, randomly switch database configurations between tests across several working instances

Penultima⬆⊠e

fitness function:

Degrade Gracefully

popular their website has become and want to make sure the site performance d

Penultima⬆⊠e

fitness function:

Maintaining Good Throughput

heir website has become and want to make sure they maintain good throughput:

Penultima⬆⊠e

fitness function:

Degrade Gracefully

- build a run-time monitor that reports on number of instances, performance p
- if performance or scalability violates the established thresholds, raise an alar

Concrete Architecture Definitions

maintainable?

Concrete Architecture Definitions

maintainable?

Degrade gracefully versus Maintain Throughput ?

Penultima⬆⊠e

fitness function:

RespondResponsibly

tors, who has a very fast website. PenultimateWidgets website has traditionally

Penultima⬆⊠e

fitness function:

RespondResponsibly

- Spin up a build in an integration environment with representative data
- Use an automated testing tool (such as Selenium or Watir) to automate visitin
- Time the rendering of the page (using a tool such as Watchtower)
- Fail the build if the response time exceeds the threshold

Penultima⬆⊠e

fitness function:

Monitor All The Things

PenultimateWidgets wants to ensure that Java and .NET applications support monitoring.

Penultima⬆⊠e

fitness function:

Monitor All The Things

- create a test during the pre-flight building of the machine image to ensure monitoring ports are listening properly
- fail the build if any development team skipped or mis-configured monitoring

Breaking Bad Habits

Penultima⬆⊠e

fitness function:

No More View Models

PenultimateWidgets has a group of developers who cut their teeth on mainframe programming, where all work is done via a terminal. They have (mostly) switched to modern web development, but architects find that they often create models and workflows directly in web pages rather than support a cleaner MVC separation, and they would like to discourage this behavior.

Penultima⬆⊠e

fitness function:

No More View Models

- use PMD (https://pmd.github.io/) or Language server protocol (https://langserver.org/) to build a tool that scans which classes developers instantiate in view code
- build a list of restricted model and controller packages
- if any view code instantiates a class from the restricted list, fail the build

Penultima⬆⊠e

fitness function:

Deploy All The Things

PenultimateWidgets has gotten tired of each application requiring an elaborate environment setup to deploy and run, and has decided that all new applications must be deployed within Docker containers. The PenultimateWidgets architects want to ensure that all applications follow the same standard deployment guidelines.

Penultima⬆⊠e

fitness function:

Deploy All The Things

Build a monitor that runs within the developers' staging area that inspects each running service to ensure that it is running within a container and supports common operational concerns such as logging, monitoring, and service discovery.

Penultima⬆⊠e

fitness function:

Deploy All The Things

Build a monitor that runs within the developers' staging area that inspects each running service to ensure that it is running within a container and supports common operational concerns such as logging, monitoring, and service discovery.

Make New Habits

Penultima⬆⊠e

fitness function:

Discover All The Things

PenultimateWidgets publishes an API for third-party, business-to-business integrations. To make it easier for clients to consume their services, they have instituted a convention that every endpoint delivers a documentation string when called with the additional parameter '&docs=true'. PenultimateWidgets architects want to ensure that all published endpoints include this documentation string.

Penultima⬆⊠e

fitness function:

Discover All The Things

Developers build a continuous monitor that calls each method in the published API at random from a dictionary, checking that each endpoint documentation invocation delivers an HTML page and not an HTML 404.

Penultima⬆⊠e

fitness function:

Install All The Things

PenultimateWidgets distributes a desktop application to select clients that facilitates coordination between business partners. The application is written in C#, using .NET desktop libraries. In the past, PenultimateWidgets has exhibited problems with the installer not working correctly for some edge cases, and they want to ensure that their clients have a good installation experience.

Penultima⬆⊠e

fitness function:

Install All The Things

- As part of the deployment pipeline, provision a pristine Windows environment for each supported version
- Install the new build and run the installation routine
- Smoke test select parts of the application to ensure that all components installed correctly.

Aggressive Back Porting

Version X

Version X+1

Aggressive Back Porting

Version X

Version X+1

Penultima⬆⊠e

fitness function:

Break on Upgrade

PenultimateWidgets builds a number of applications in Ruby on Rails, which has an active release schedule. Occasionally, developers really want a feature that appears in version X+1, where PenultimateWidgets is currently on version X. To acquire the desired functionality, developers back-port the new features from X + 1 into X. However, when PenultimateWidgets finally does upgrade to X + 1, the back ported features are almost certain to break because of incompatibilities.

Penultima⬆⊠e

fitness function:

Break on Upgrade

- Add a unit test to the code base that checks the version number of the framework. If the framework version it higher than the expected one, fail the build with a warning that back-ported features will likely not work.
- Catalog all the back-ported features in the unit test to make it easy to identify them when it becomes time to replace them.

Penultima⬆⊠e

fitness function:

Upgrade All The Things

PenultimateWidgets has a bad habit of waiting too long to update core libraries and frameworks they depend upon for development. Waiting past several versions makes the eventual upgrade quite painful, and they have resolved to perform upgrades in a more timely manner.

Penultima⬆⊠e

fitness function:

Upgrade All The Things

- Use Dependabot to flag new major versions of frameworks and libraries.
- When a library updates, make an entry in a build database and start a 3-month clock.
- If the library or framework in question hasn't been updated in 3 months, fail the build.

Penultima⬆⊠e

fitness function:

Depend on Dependencies

PenultimateWidgets has recently been burned by a Maven update that broke several applications yet wasn't detected until production. Architects and business people want to make sure this doesn't happen again.

Penultima⬆⊠e

fitness function:

Depend on Dependencies

Use a deployment pipeline to trigger new builds and test suite execution any time any artifact of the project changes, including external dependencies. If the automatic build breaks, a developer must locate the problem and restore automatically.

OR

Use Dependabot (https://dependabot.com/)

Penultima⬆⊠e

Penultima⬆⊠e

Penultima⬆⊠e

incremental

Penultima⬆⊠e

Evolving Routing

< >

<home> <catalog><item>

Evolving Routing

Penultima⬆⊠e

Evolving Routing

Penultima⬆⊠e

Evolving Routing

Penultima⬆☒e

Evolving Routing

Routes

RouteContext

Penultima⬆⊠e

Experimenting with New Markets

Penultima⬆⊠e

fitness function:

MaintainTheMock

PenultimateWidgets has started experimenting with flashing the chips on their 'smart' widgets during the manufacturing process, allowing them to make changes in hardware-based code more easily and quickly. However, they must make sure that the mock version of the hardware API the developers use is always up to date with the real API.

Penultima⬆⊠e

fitness function:

MaintainTheMock

Build a fitness function that matches the mock API method signatures, parameters, and types to the real API on the hardware; fail the build if the team has allowed it to get out of sync.

govern

Origin

GREEK	LATIN	OLD FRENCH	
kubernan →	gubernare →	governer →	govern
to steer	to steer, rule		*Middle English*

—to control, direct, or strongly influence the actions and conduct of

—to exert a determining or guiding influence in or over

—to hold in check

monolithic governance

decentralized governance

govern

Origin

GREEK	LATIN	OLD FRENCH	
kubernan →	gubernare →	governer →	govern
to steer	to steer, rule		*Middle English*

—to serve as a precedent or deciding principle for

20th Century Governance

21st Century Governance

>

guidance over cost savings

“Goldilocks” Governance

Choose technology stacks appropriate to problem scale.

Penultima⬆⊠e

Deployment Pipeline

Penultima⬆⊠e

fitness function:

Zero Day Security Check

PenultimateWidgets management has become concerned about zero-day exploits in open source libraries and has tasked the security team with coming up with a way of ensuring that projects do not use known compromised versions.

Penultima⬆⊠e

fitness function:

Zero Day Security Check

- build a common fitness function into every deployment pipeline for the secu
- have the security fitness function check version numbers of deployed librari
- if a tainted version is in use, fail the build and notify the team

Penultima⬆⊠e

Penultima⬆⊠e

fitness function:

Doc Sync With API

PenultimateWidgets publishes an API for external vendors to check on order status, availability, and other B2B perks. However, the external partners rely on the documentation of the API, which sometimes becomes outdated when developers add new functionality. The PenultimateWidgets architects want to ensure that documentation is always up to date, especially method signatures and data formats.

Penultima⬆⊠e

fitness function:

Doc Sync With API

- Parse the code to look for public method signatures
- Parse the documentation looking for <code> tags, which surround method signatures and parameters
- Ensure that each tagged element includes a description
- Add a Publication Date value to the public website; after new deployments, check to make sure the deployment time matches the Publication date

Penultima⬆⊠e

fitness function:

Sell The Platform

PenultimateWidgets is in the process of rebuilding their core platform, incorporating some key features that will appeal to others in a similar problem space. PenultimateWidgets plans to sell their platform to other companies, but worry that the differentiating factors (such as performance, scale, and elasticity) will suffer as clients start making changes to the platform, ultimately harming PenultimateWidgets platform's reputation. How can they sell their platform and still ensure that the critically important characteristics remain unchanged?

Penultima⬆⊠e

fitness function:

Sell The Platform

- PenultimateWidgets sells not only the platform but the running deployment pipeline for the platform as well. PenultimateWidgets engineers make sure that all the fitness functions run in the new client's environment.
- PenultimateWidgets will continue to certify the platform as long as all their fitness functions continue to execute successfully.

Penultima⬆⊠e

Penultima⬆⊠e

Penultima⬆⊠e

alien artifacts!

Penultima⬆⊠e

alien artifacts!

integration architecture

(Forced) Integration

(Forced) Integration

(Forced) Integration

(Selective Use of) ESB

“Legacy in a Box” Pattern

puBLisHinG goVeRnANce

Build Your Own Technology Radar

ThoughtWorks®

Technology Radar

OUR THOUGHTS ON JAVASCRIPT, APIS, CONWAY'S LAW, RE-DECENTRALIZATION AND MUCH MORE

JULY 2014

thoughtworks.com/radar

THE RADAR

TECHNIQUES

ADOPT
1 Capturing client-side JavaScript errors
2 Continuous delivery for mobile devices
3 Mobile testing on mobile networks
4 Segregated DOM plus node for JS Testing
5 Windows infrastructure automation

TRIAL
6 Capture domain events explicitily
7 Client and server rendering with same code
8 HTML5 storage instead of cookies
9 Instrument all the things
10 Masterless Chef/Puppet
11 Micro-services
12 Perimeterless enterprise
13 Provisioning testing
14 Structured Logging

ASSESS
15 Bridging physical and digital worlds with simple hardware
16 Collaborative analytics and data science
17 Datensparsamkeit
18 Development environments in the cloud
19 Focus on mean time to recovery
20 Machine image as a build artifact
21 Tangible interaction

HOLD
22 Cloud lift and shift
23 Ignoring OWASP Top 10
24 Siloed metrics
25 Velocity as productivity

PLATFORMS

ADOPT
26 Elastic Search
27 MongoDB
28 Neo4J
29 Node.js
30 Redis
31 SMS and USSD as a UI

TRIAL
32 Hadoop 2.0
33 Hadoop as a service
34 OpenStack
35 PostgreSQL for NoSQL
36 Vumi

ASSESS
37 Akka
38 Backend as a service
39 Low-cost robotics
40 PhoneGap/Apache Cordova
41 Private Clouds
42 SPDY
43 Storm
44 Web Components standard

HOLD
45 Big enterprise solutions
46 CMS as a platform
47 Enterprise Data Warehouse

THE RADAR

TOOLS

ADOPT
48 D3
49 Dependency management for JavaScript

TRIAL
50 Ansible
51 Calabash
52 Chaos Monkey
53 Gatling
54 Grunt.js
55 Hystrix
56 Icon fonts
57 Librarian-puppet and Librarian-Chef
58 Logstash & Graylog2
59 Moco
60 PhantomJS
61 Prototype On Paper
62 SnapCI
63 Snowplow Analytics & Piwik

ASSESS
64 Cloud-init
65 Docker
66 Octopus
67 Sensu
68 Travis for OSX/iOS
69 Visual regression testing tools
70 Xamarin

HOLD
71 Ant
72 Heavyweight test tools
73 TFS

LANGUAGES & FRAMEWORKS

ADOPT
74 Clojure
75 Dropwizard
76 Scala, the good parts
77 Sinatra

TRIAL
78 CoffeeScript
79 Go language
80 Hive
81 Play Framework 2
82 Reactive Extensions across languages
83 Web API

ASSESS
84 Elixir
85 Julia
86 Nancy
87 OWIN
88 Pester
89 Pointer Events
90 Python 3
91 TypeScript
92 Yeoman

HOLD
93 Handwritten CSS
94 JSF

Thoughtworks, Inc.

ThoughtWorks®

BUILD YOUR INTERACTIVE RADAR

How to build your own radar

Once you've ran your radar creation exercise, you'll want to share the output. Our "radar visualization" service generates an interactive version of your technology radar.

This service is inspired by the ThoughtWorks Technology Radar. It's currently in beta and provides just enough functionality to serve its purpose. You can give us feedback, suggest feature improvements and get updates here.

See a demo

How to use

All you need to visualize your radar is a ***public Google Sheet*** containing your

Hear more about building your own radar

First Name: *

Last Name: *

Email Address: *

Role: *

https://info.thoughtworks.com/visualize-your-tech-strategy-guide.html

Example Technology Radar

File Edit View Insert Format Data Tools Add-ons Help All changes saved in Drive

	A	B	C	D	E
1	***name***	***ring***	***quadrant***	***isNew***	***description***
2	Babel	adopt	tools	FALSE	This is the description. You can use basic html such as the stro and insert <a href="https://www.thoughtworks.com">anchor links to docu
3	Consul	adopt	tools	FALSE	This is the description. You can use basic html such as the stro and insert <a href="https://www.thoughtworks.com">anchor links to docu
4	Grafana	adopt	tools	FALSE	This is the description. You can use basic html such as the stro and insert <a href="https://www.thoughtworks.com">anchor links to docu
5	Apache Kafka	trial	tools	TRUE	This is the description. You can use basic html such as the stro and insert <a href="https://www.thoughtworks.com">anchor links to docu
6	Espresso	trial	tools	TRUE	This is the description. You can use basic html such as the stro and insert <a href="https://www.thoughtworks.com">anchor links to docu
7	Consumer-driven co	adopt	techniques	FALSE	This is the description. You can use basic html such as the stro and insert <a href="https://www.thoughtworks.com">anchor links to docu
8	Bug bounties	trial	techniques	TRUE	This is the description. You can use basic html such as the stro and insert <a href="https://www.thoughtworks.com">anchor links to docu
9	Client-directed query	assess	techniques	FALSE	This is the description. You can use basic html such as the stro and insert <a href="https://www.thoughtworks.com">anchor links to docu
10	Container security s	assess	techniques	FALSE	This is the description. You can use basic html such as the stro and insert <a href="https://www.thoughtworks.com">anchor links to docu

ThoughtWorks®

BUILD YOUR RADAR

Once you've created your radar, you can use this service to generate an interactive version of your Technology Radar. Not sure how? Read this first.

Enter the URL of your public google sheet below...

e.g. https://docs.google.com/spreadsheets/d/1--_uLSNf/pubhtml

Build my radar

Need help?

https://github.com/thoughtworks/build-your-own-radar

Features Explore Pricing

This repository Search

Sign in or Sign up

thoughtworks / build-your-own-radar

Watch 54 Star 107 Fork 72

Code Issues 7 Pull requests 0 Projects 0 Pulse Graphs

A library that generates an interactive radar, inspired by http://thoughtworks.com/radar/

radar thoughtworks opensource

238 commits 4 branches 0 releases 14 contributors AGPL-3.0

Branch: master New pull request Find file Clone or download

NivedhaSenthil committed on GitHub Merge pull request #13 from shahadarsh/column_name_fix ... Latest commit e95bf09 on Jan 9

scripts	updated publish examples scripts to fetch from the right directory	2 years ago
spec	Arun: Accept sheet URL in various formats	3 months ago
src	Fix for columns using Capital letters	2 months ago
.gitignore	re-add examples to git ignore	5 months ago
.istanbul.yml	adding coverage to new code setup	5 months ago
LICENSE.md	Update LICENSE.md	4 months ago
README.md	Update README.md	2 months ago

https://github.com/thoughtworks/build-your-own-radar

puBLisHinG goVeRnANce

Penultima⬆⊠e

Setting WIP limits for experiments

enterprise architecture transparency

Broadcasting R&D

Penultima⬆☒e

Penultima⬆☒e

Penultima⬆☒e

ideation
selected experiments:
pivot
fold
double down
portfolio of experiments

Penultima⬆☒e

Penultima⬆⊠e

OpenUltima⬆⊠e

Penultima⬆⊠e

OpenUltima⬆⊠e

Evolutionary architecture provides the substrate for 21st century growth.

Fitness Function Guidelines

most fitness function are local

obal only when universally applie

No architecture states!

Fitness Function Guidelines

developers must collaborate

No architecture states!

Fitness Function Guidelines

utomation but rely on manual whe

automation =

when needed

Architecture Defined

architecture =

+

prioritization

cost

http://evolutionaryarchitecture.com

iMpLemENtiNg eVoLuTiONaRy ARcHitECtuREs

@neal4d
nealford.com

@rebeccaparsons

@patkua

Penultima⬆⊠e

Penultima⬆⊠e

OpenUltima⬆⊠e