Faster, Cheaper Identity Management through Loose Coupling – the LIMA Approach

Ganesh Prasad

Speaker Bio

Ganesh Prasad

Sydney-based Architect

25 years in IT, 9+ years in “Shared Services”

Author of white papers on

- Presentation (“Life Above the Service Tier”)
- SOA (“Practical SOA for the Solution Architect”)

Initial exposure to IAM (Identity and Access Management) systems at a Big Four Australian bank

Implemented a loosely-coupled IAM system at a large insurance company (as documented in this InfoQ eBook)

Currently working to implement IAM at a major telecom company

A Word about Shared Services

- Also called “Enterprise Utilities”

- Not domain-Specific, used by multiple business units, not owned by any of them

E.g., **Identity and Access Management (IAM)**, Content Management, Integration, Communication Gateways, Document Composition, etc.

- Enterprise funding or “first project pays”?

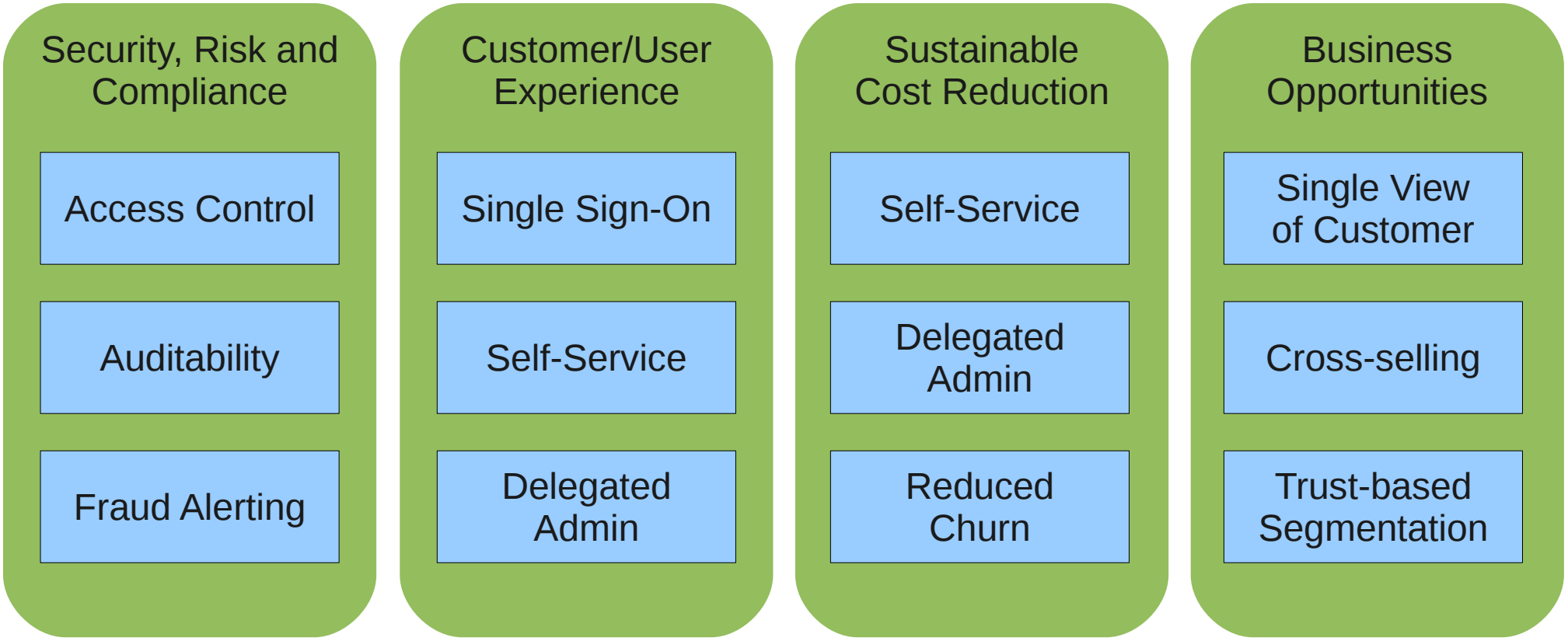
- The biggest technology challenge is to break up a shared service into smaller chunks such that:

1. each chunk delivers value to a business unit
2. the business unit will therefore be willing to fund that chunk
3. the chunks may be delivered in any order
4. the chunks will play well together to eventually form the shared service

- Loose coupling is the only way.

Aligning IAM with Business Benefits

Every business unit wants something different from IAM.



Build a different business case for each chunk of identity management based on what the respective business unit is willing to pay for.

That makes the loosely-coupled approach viable.

What is Identity?

A simple, informal definition:

A unique identifier and a set of attributes for an entity that are meaningful in a given context.

Identity only makes sense within a context

Mr. Smith

Is CTO of Acme Corp

Is a member of the Qantas Frequent Flyer program

Subscribes to Fortune magazine

John

Mobile number is 0499 999 999.

Has a friendly labrador called Spiffy

Is interested in Science Fiction

Dad

Can help with math homework

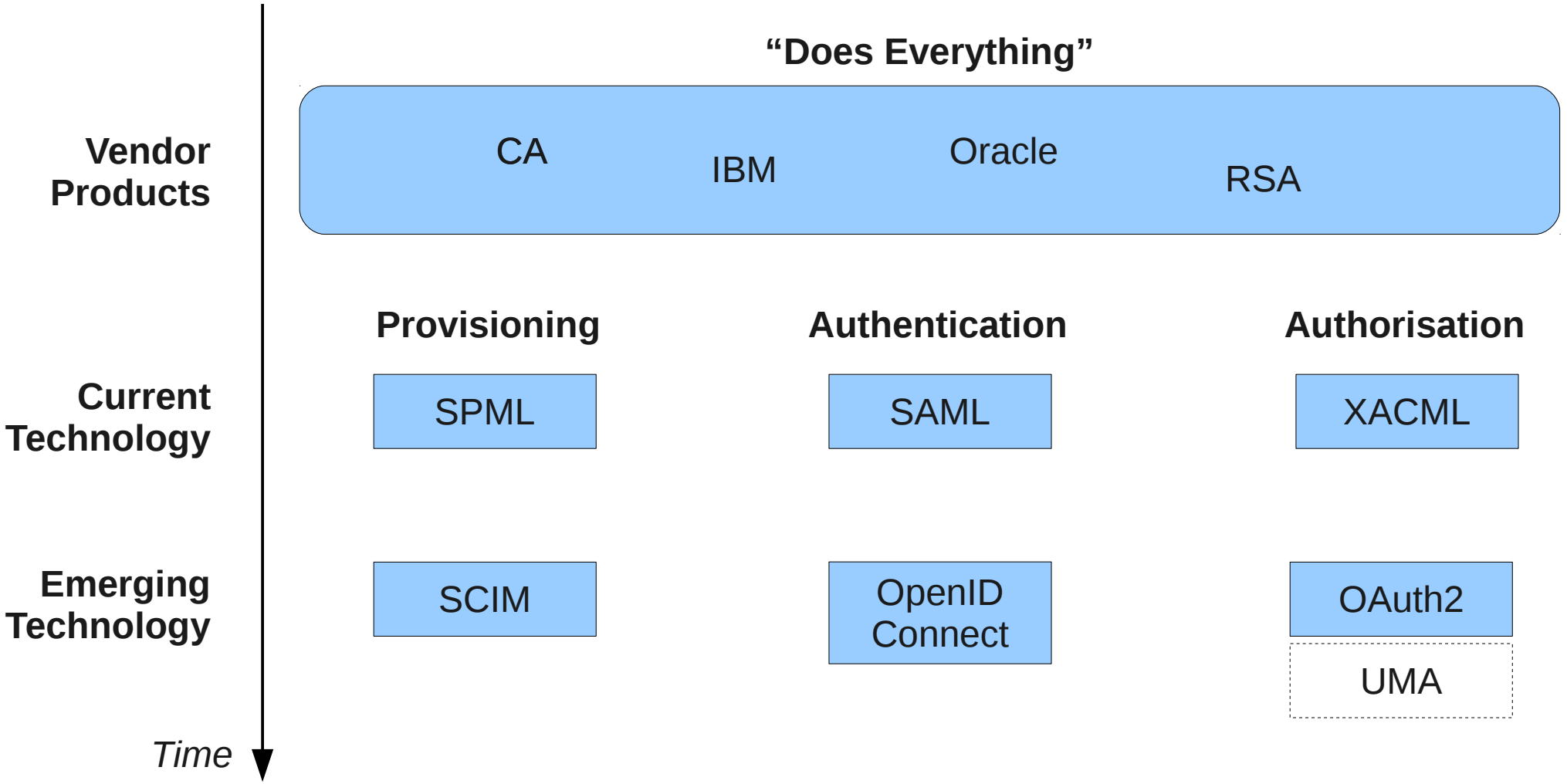
Is grumpy in the morning

Isn't interested in my piano playing

Attributes, including identifiers, aren't usually meaningful outside a context.

Identifiers may be associated across domains.

Identity and Access Management - More than just technology



Technology is cool, of course, but it's not the only aspect of IAM...

…there's also Data!

(If you're tightly-coupled on data, you're tightly-coupled, period.)

Identity is fundamentally a data problem...

… not a technology problem.

So the good news is that an IAM solution doesn't have to cost very much. (It's just design!)

But Identity is also a subtle concept.

So the bad news is it's very easy to get it wrong...

Corporate Decision-Making 101

- I don't know driving and I don't have a driver's licence
- I need to get from Point A to Point B

Therefore (drumroll),

- I think I should buy a BMW because my expert friends tell me it's a great car!

The “corporate” approach

Statement of problem: I need a car.

Ask the experts which car they recommend

Shortlist – BMW, Benz or Lexus?

I think the BMW is the best.

Buy the BMW. (The biggest problem is *&$%$#* funding!)

Drive the car.

Crash.

I wonder what happened. Perhaps I should have bought the Merc instead.

The sensible approach

Statement of problem: I need to get from Point A to Point B.

Ask the stupid questions: Can I take public transport? Can I join a car pool?

No? OK, looks like I'll have to drive my own car.

Recognise the immediate problem: I don't know how to drive!

Learn to drive. Pass the driving test. Get the licence.

Buy a car within my budget. A second-hand Hyundai Accent? Fine.

Drive from Point A to Point B. (Boring. Unglamorous.)

Throw the money saved at the mortgage. Move on to solving the other problems in my life.

The problem with the sensible approach

Nobody gets to see a BMW parked in my driveway!

Arguments against the sensible approach

• Trust the experts; analysts have studied these markets and technologies in depth; vendors understand industry best practice

• Don't roll your own; don't reinvent the wheel; “Buy before build”; outsource non-core competencies

• Don't mess with security; you're not a security expert; the auditors won't like you designing your own Identity Management system

Arguments for the sensible approach

- By all means, outsource non-core competencies, but don't outsource your brains.

- Architecture first, then product

- Identify capability gaps, then plug them in a pragmatic way.

But how do we do that? Ask Harry Potter.

“But why couldn't Quirrell touch me?” […] “Love, Harry. Love.”

Time to whip out some Old Magic, then.

Old Architecture Magic: Cohesion and Coupling

The terms "cohesion" and "coupling" were first introduced in "Structured Design" - a 1974 paper by WP Stevens, GJ Myers and LL Constantine

Cohesion:

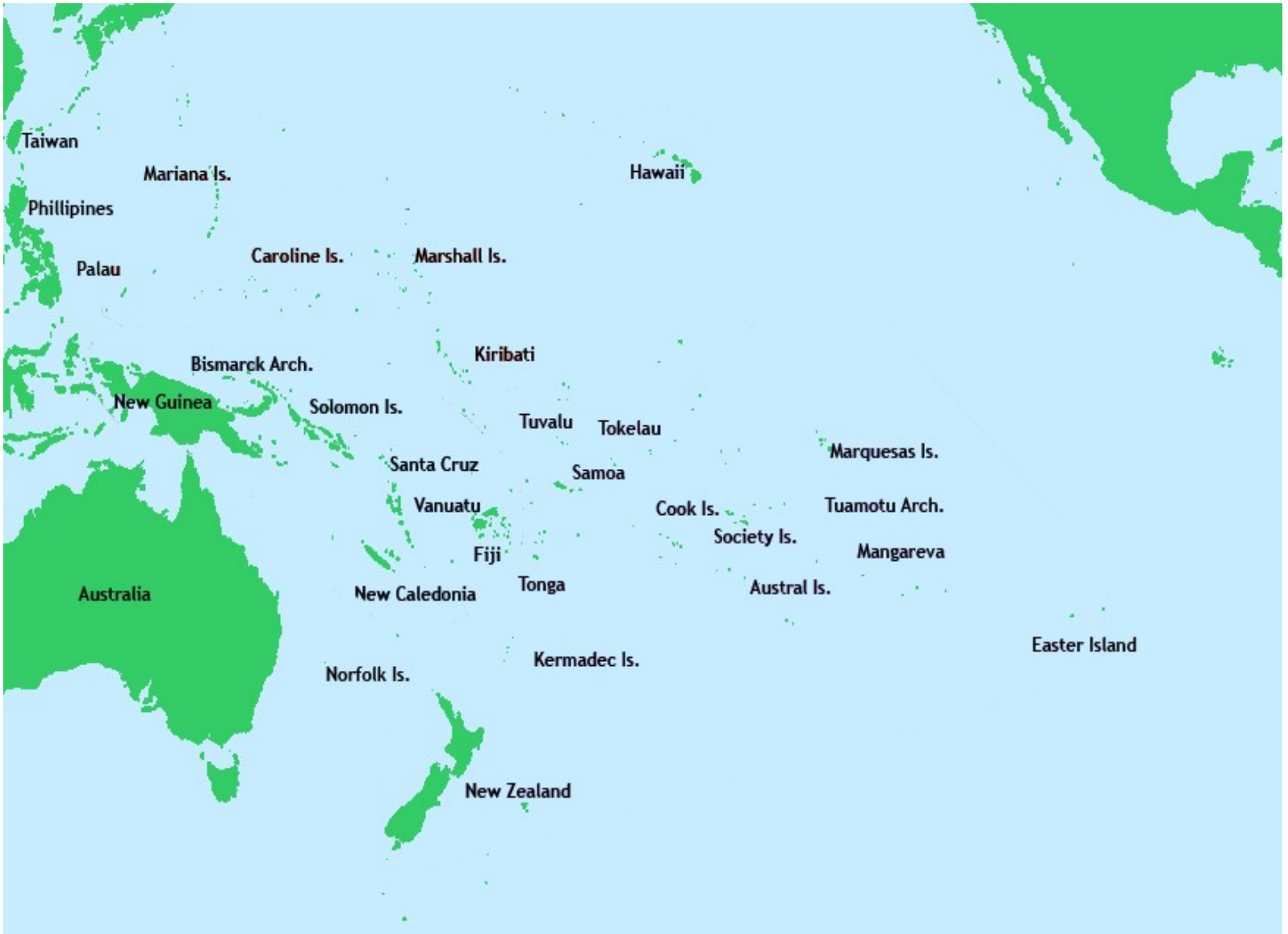
"Cohesion refers to how closely all the routines in a class or all the code in a routine support a central purpose" - Steve McConnell

i.e., things that belong together should go together.

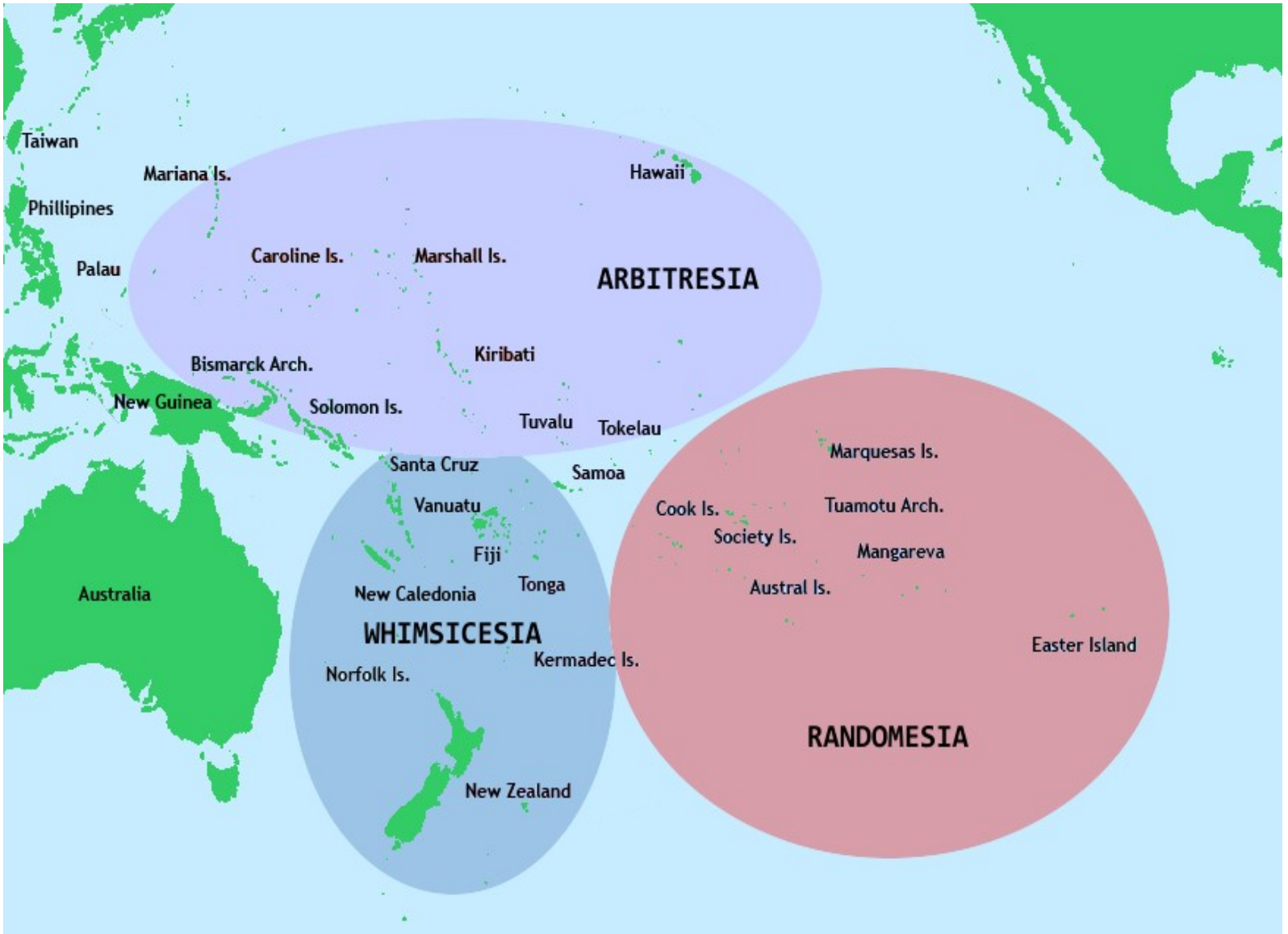
Coupling:

"If two modules communicate, they should exchange as little information as possible" - Bertrand Meyer

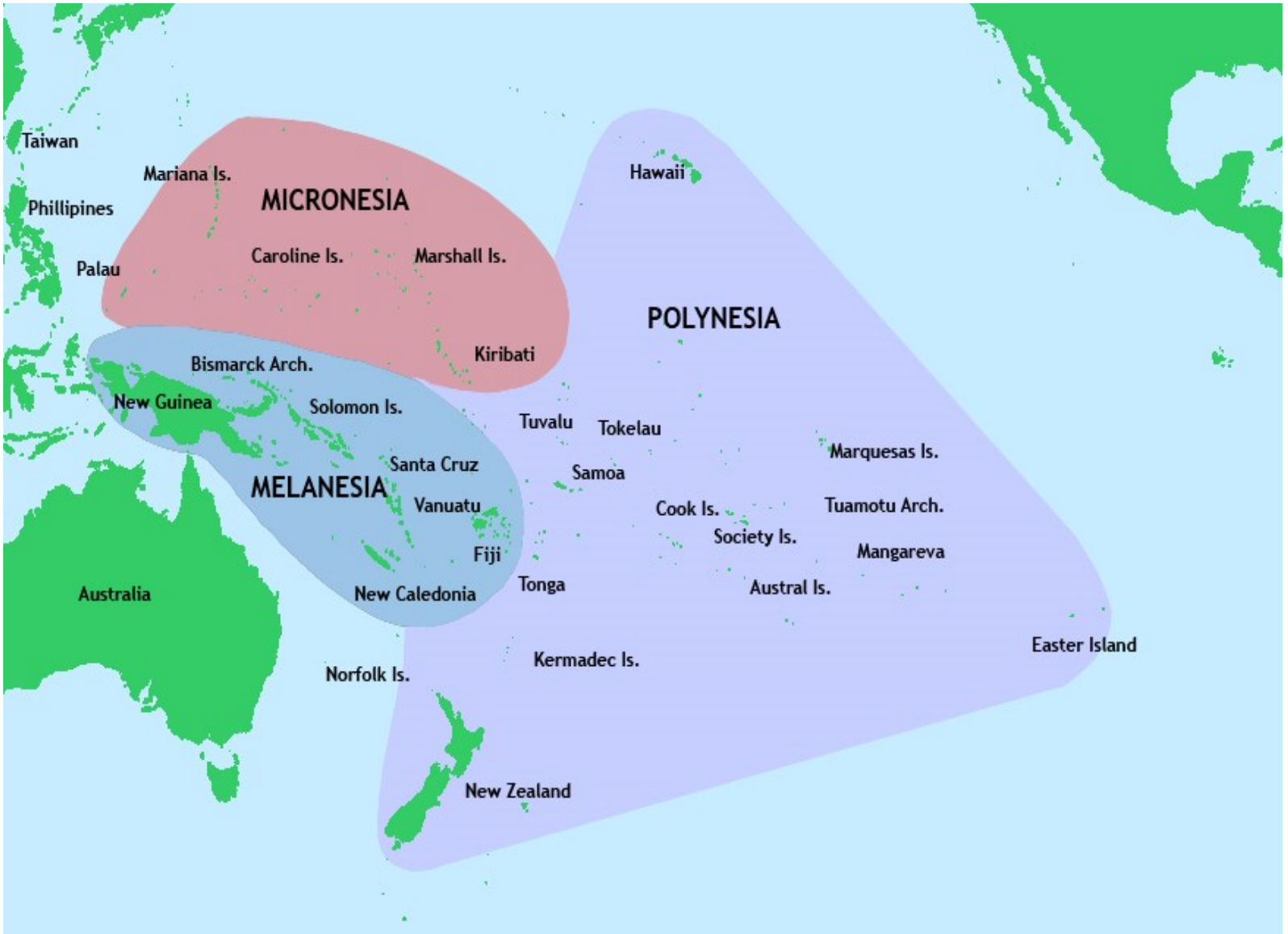
Cohesion Problem – Group these islands



Wrong Answer



Right Answer



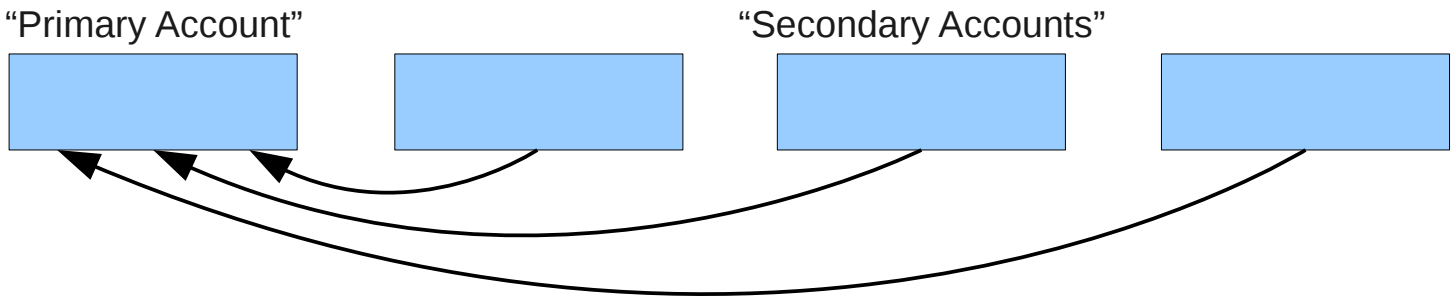
The LIMA* Approach (Cohesion and Coupling Applied to Identity Management)

* Loosely-coupled/Lightweight/Low-cost Identity Management Architecture

Lesson #1 – No Surrogates!

Example: Banks have traditionally been account-centric. Until recently, they didn't have a view of what a “customer” was!

Some early attempts at getting a customer-oriented view out of account-oriented systems looked like this:

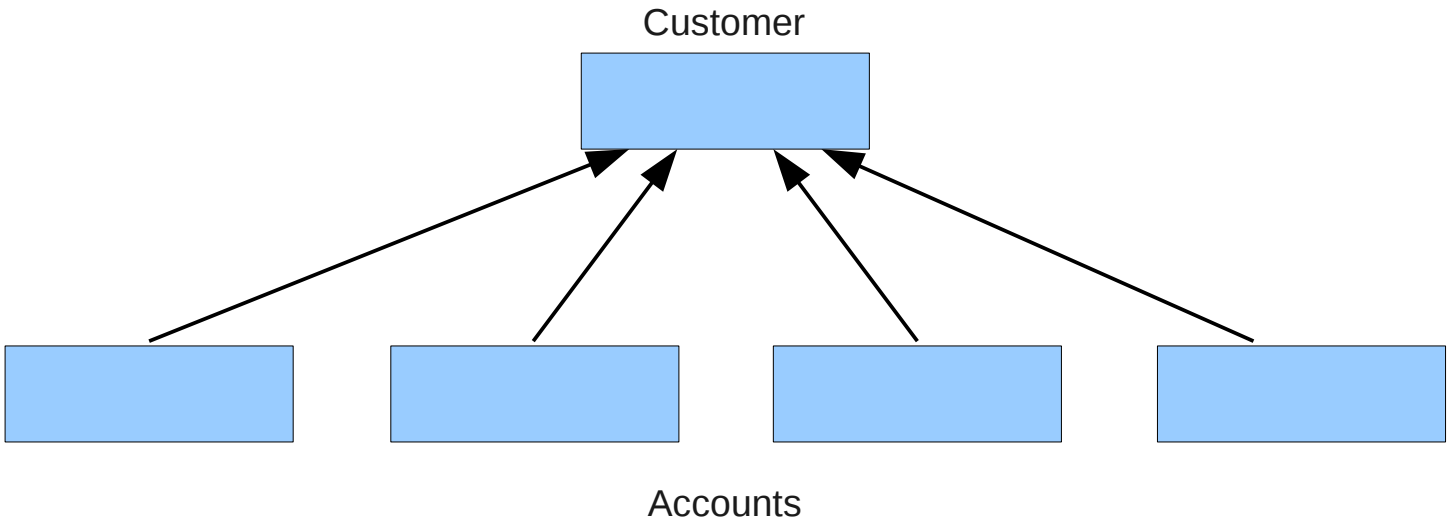


That sort of worked, except when the customer wanted to close the “primary account”...

The relationship between a customer and an account is a “HAS-A” relationship, not an “IS-A” relationship. So substituting an account for a customer isn't ever going to work. Cohesion fail!

Lesson #1 cont'd

Then banks bit the bullet and started to do what they should have done from the start:



That's Lesson #1:
When you want to talk about something, make it a first-class entity and don't try and use surrogates.

Mixing up concepts is one of the worst forms of tight coupling.

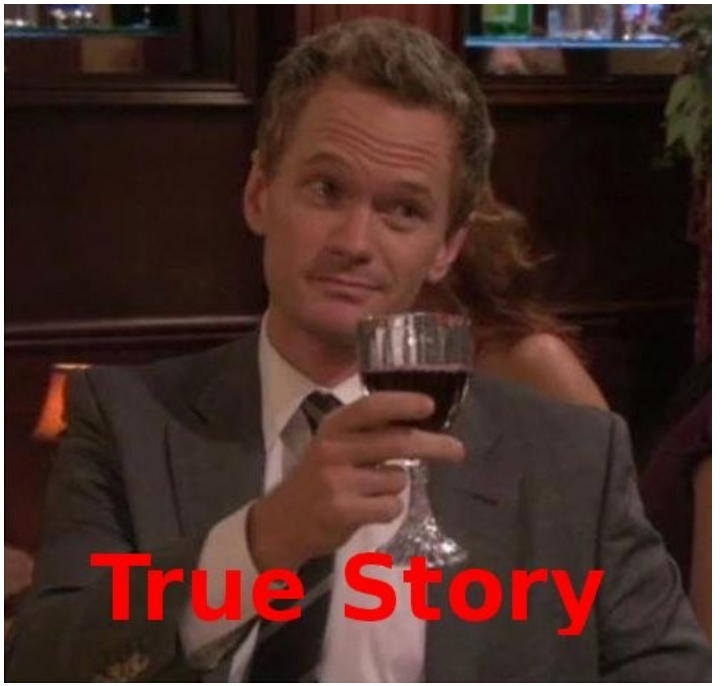
Lesson #2 – Identifiers should be meaning-free

What is a good identifier for a person (that an Identity Management system can use)?

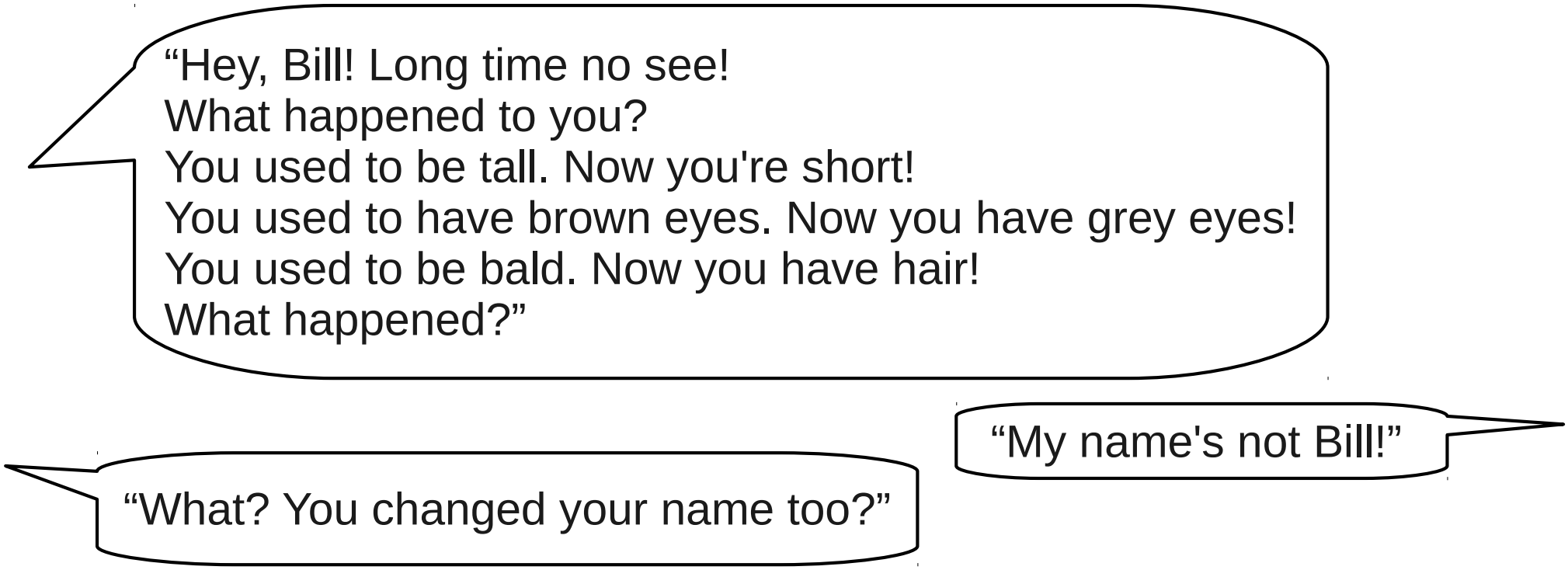
a) Name (First Name + Last Name)
b) Primary Email Address
c) Social Security Number or equivalent
d) A combination (First Name + Last Name + DoB + Address)
e) None of the above

Correct answer: e

We know of a company that chose “email address” as the identifier for their customers, then had to jump through hoops to handle the case when a customer changed their email address.



Lesson #2 cont'd



An attribute with meaning can change in value when the context changes.

A person can change their name, their passport number, their email address, etc. They can even change (i.e., correct) their date of birth.

But who they ARE doesn't change!
So *meaningful* identifiers are a form of tight coupling...

Lesson #2 – Identifiers should be meaning-free

Have a UUID!

296a19cf-fc95-4077-a539-d1e17106267a

f9f961ce-6b37-4ccd-9d14-96744cd8dd13

c28dae97-8997-46b0-8cb1-2f1d4647eef6

0fbbbfd0-8901-4989-9d54-bf0b8f958ae6

e7731005-d2b0-453d-aafe-779f48438d3a

9ac6ed26-c726-45a4-93aa-6376c750a300

a82fe650-5af7-4ed4-be4e-e6c90018c60c

b1fdfb0c-6c18-4be4-8103-11b99c7015fd

9d27d623-aaf7-4baf-84dc-a9e32027cfca

b2b10121-206b-4466-9e9c-f06e1c25ad78

Plenty more where these came from (10^{33} more...)

Advantages of UUIDs

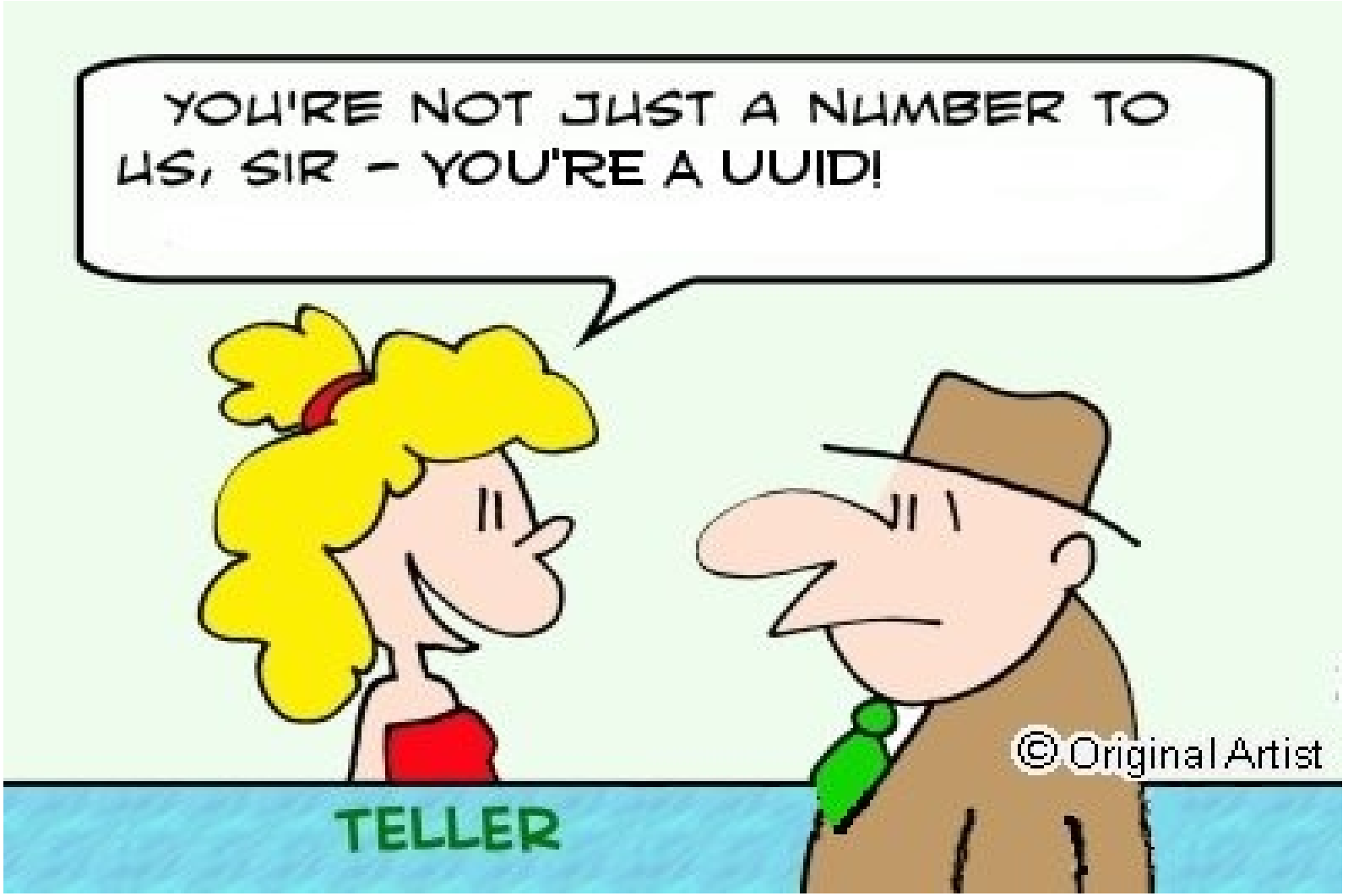
Federation-friendly – multiple systems can generate UUIDs independently without danger of conflict. No need to rely on a single authoritative source of identifiers.

Supports optimistic processes – the probability of conflicting IDs is ridiculously low. We can check for duplicates once in a while rather than at the time of entity creation.

Supports autonomy – domains can implement systems at their own pace. No need for “coordination” because the touchpoints can be reduced to opaque identifiers.

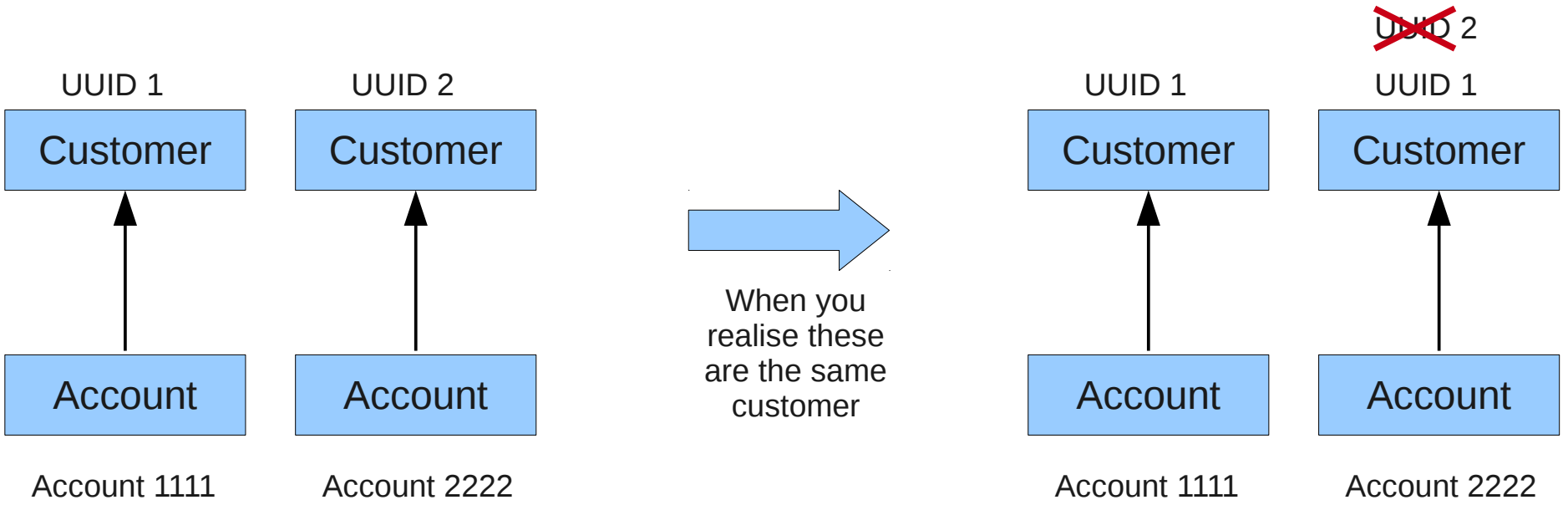
Supports incremental funding – smaller domains can be implemented independently and harmonised with each other in due course. No need for a Big Bang approach with high initial costs.

This yields better customer service than you know!



Lesson #3 – Don't expose internal identifiers

Your model of identity needs to be flexible enough to change.

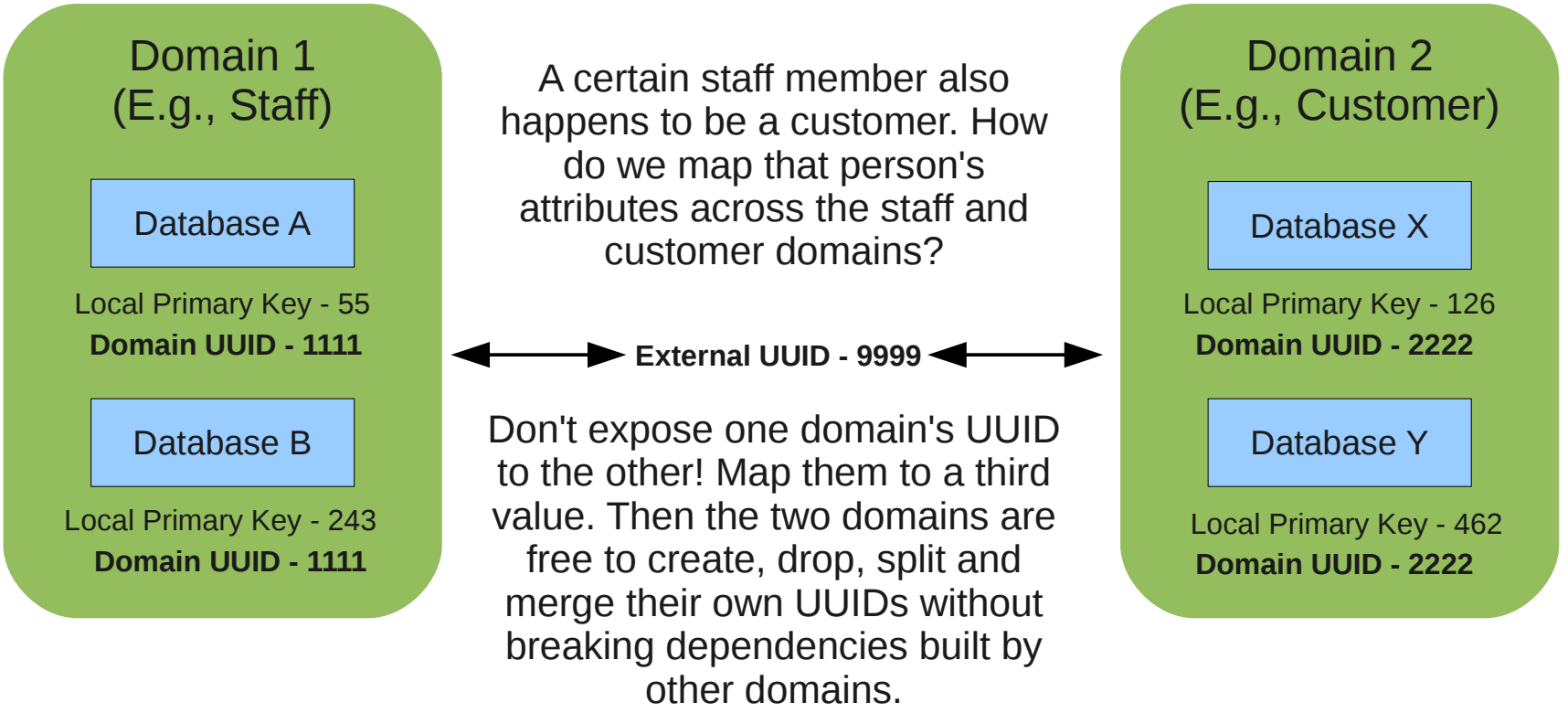


Splitting and merging identities is easier when you don't expose identifiers externally – an “eventual consistency” model.

So exposing internal identifiers to external parties is yet another form of tight coupling.

Lesson #3 Cont'd – How to Expose Identifiers

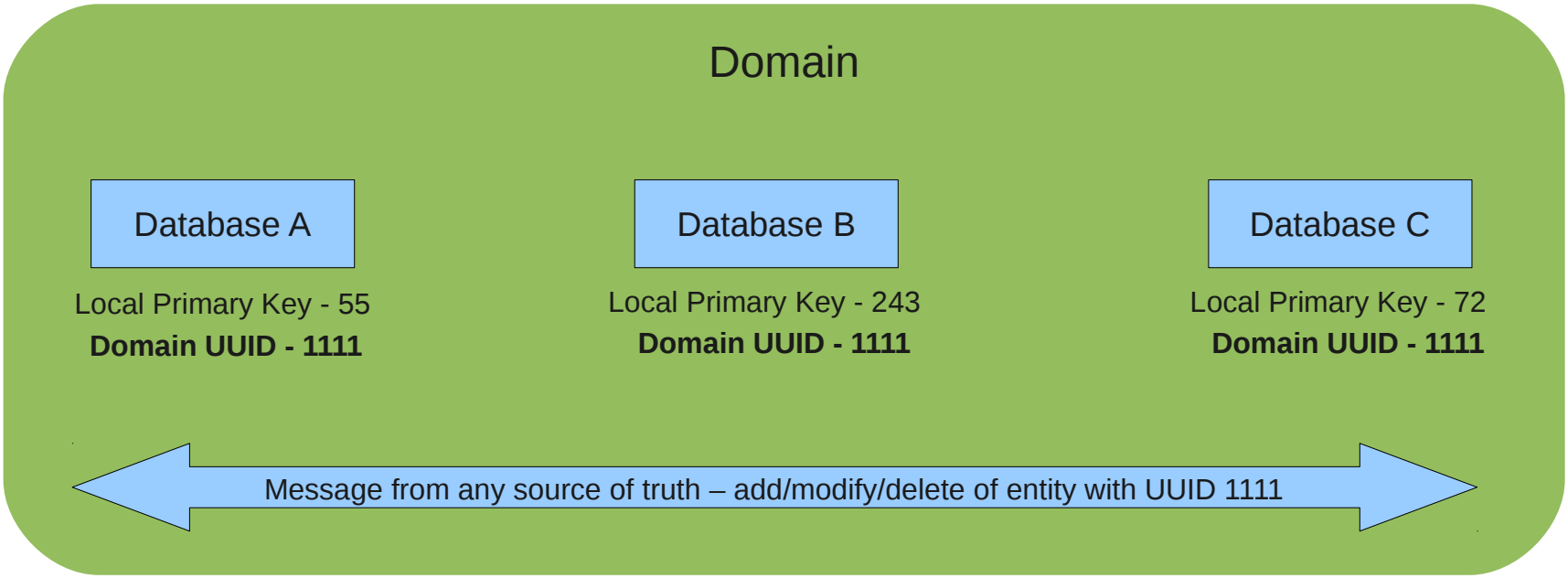
Within a domain, a UUID is the common *candidate key* that harmonises all references to a given entity across databases, regardless of the different values of its primary key within those databases.



The UUID is still private to the domain. If an entity reference must cross domains (e.g., a staff member is also a customer), it must be *translated* to an external UUID. The other domain must translate this external UUID into its private domain UUID.

Lesson #4 – Don't Unify, Harmonise!

Data will never be normalised. Duplicated data is a part of life. Get over it.



Master Data Management (MDM) principles are simple.
Never update a replica, only ever refresh it from the source of truth.
Use the UUID to reference a logical record across data stores in the domain.

Summary

Data design is more important than technology in the area of Identity and Access Management. Great technology cannot make up for bad data design.

Know what you're talking about. Bite the bullet and create first-class entities instead of sneaking by using surrogates. Model relationships correctly. The data model must be correct, complete and consistent.

Ensure that your entity identifiers are globally unique, meaning-free and externally invisible.

Use the UUID to harmonise replicas of data with the sources of truth.

Great Technology

Sound Data Model

Thank You!

Questions?