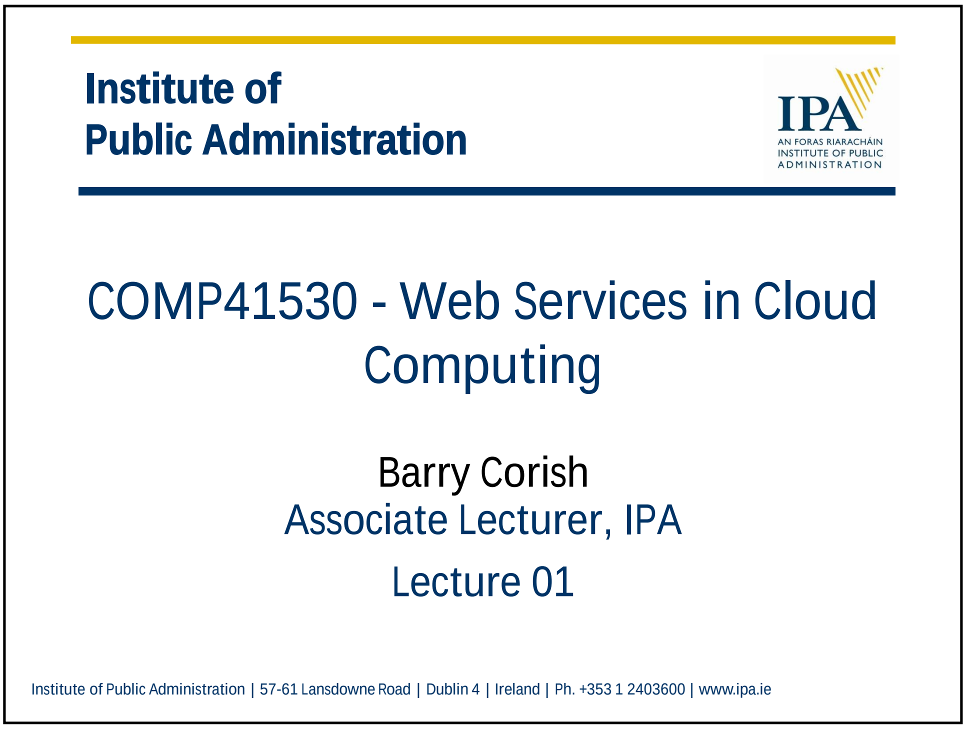
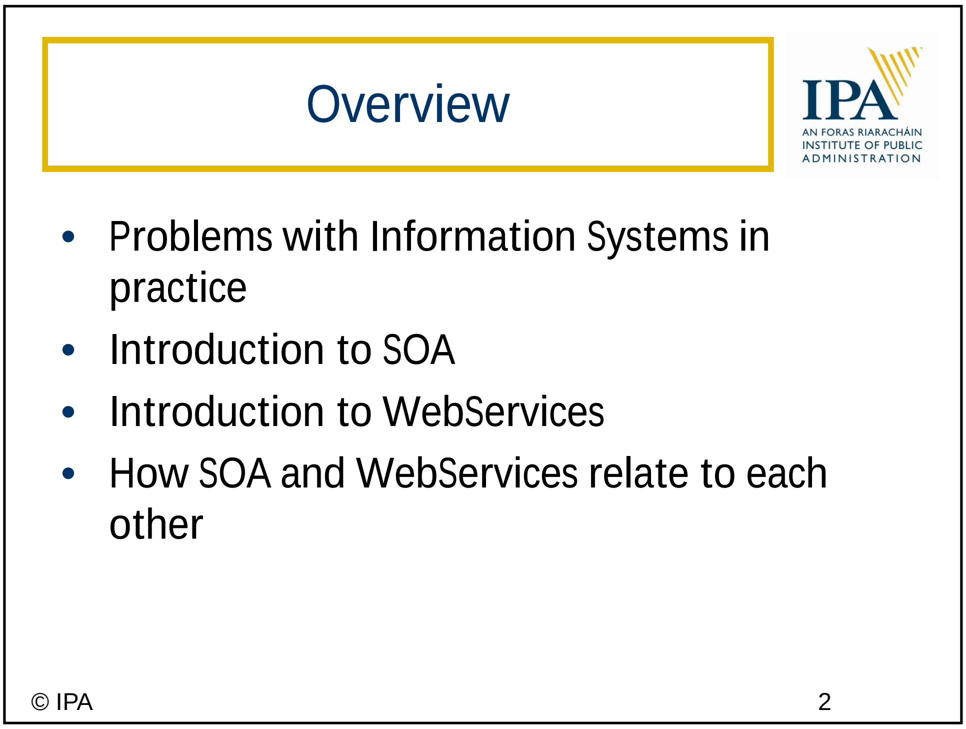


Institute of
Public Administration

COMP41530 - Web Services in Cloud Computing

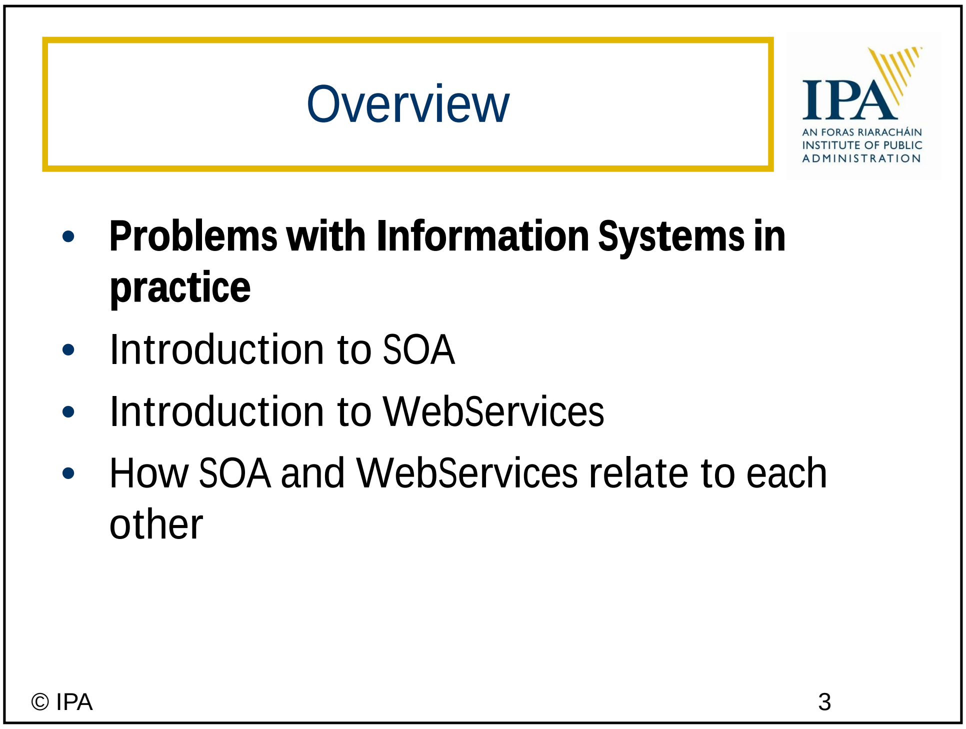
Barry Corish
Associate Lecturer, IPA

Lecture 01



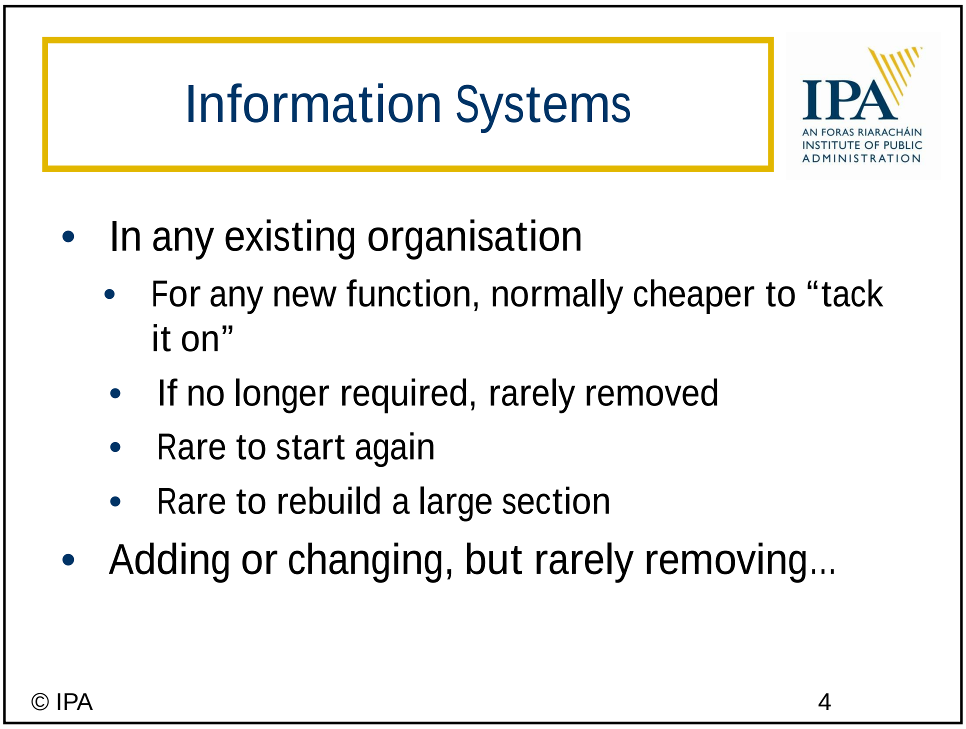
Overview

- Problems with Information Systems in practice
- Introduction to SOA
- Introduction to WebServices
- How SOA and WebServices relate to each other



Overview

- **Problems with Information Systems in practice**
- Introduction to SOA
- Introduction to WebServices
- How SOA and WebServices relate to each other



Information Systems

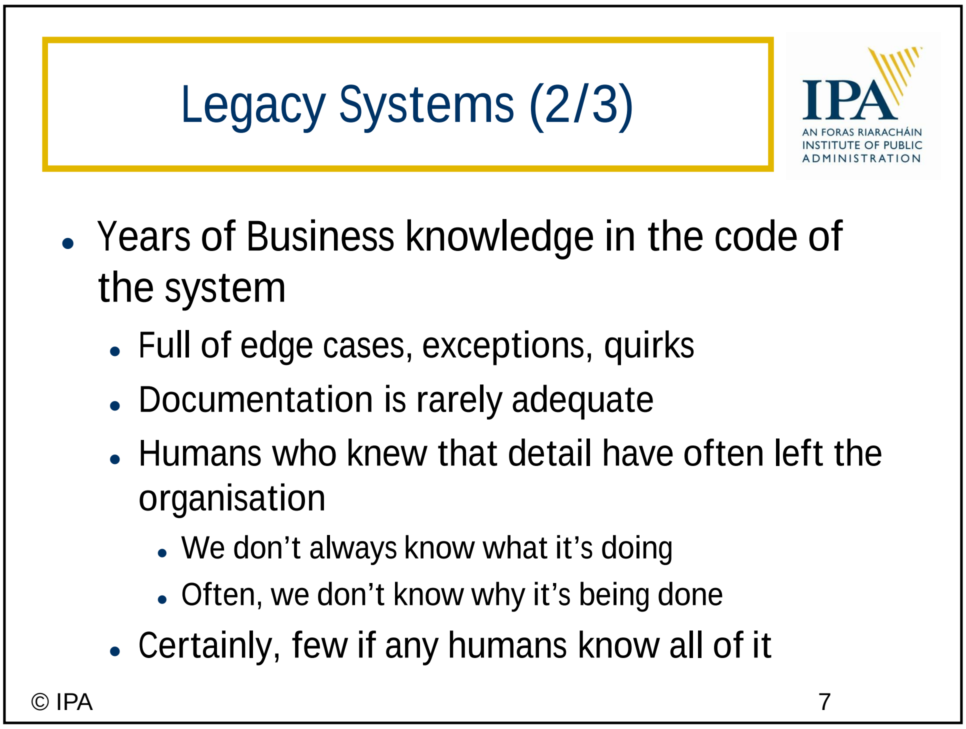
- In any existing organisation
  - For any new function, normally cheaper to “tack it on”
  - If no longer required, rarely removed
  - Rare to start again
  - Rare to rebuild a large section
- Adding or changing, but rarely removing…

Information Systems

- Over time, Information Systems tend to become:
  - larger
  - more complicated
  - more embedded in an organisation
  - harder to change or replace

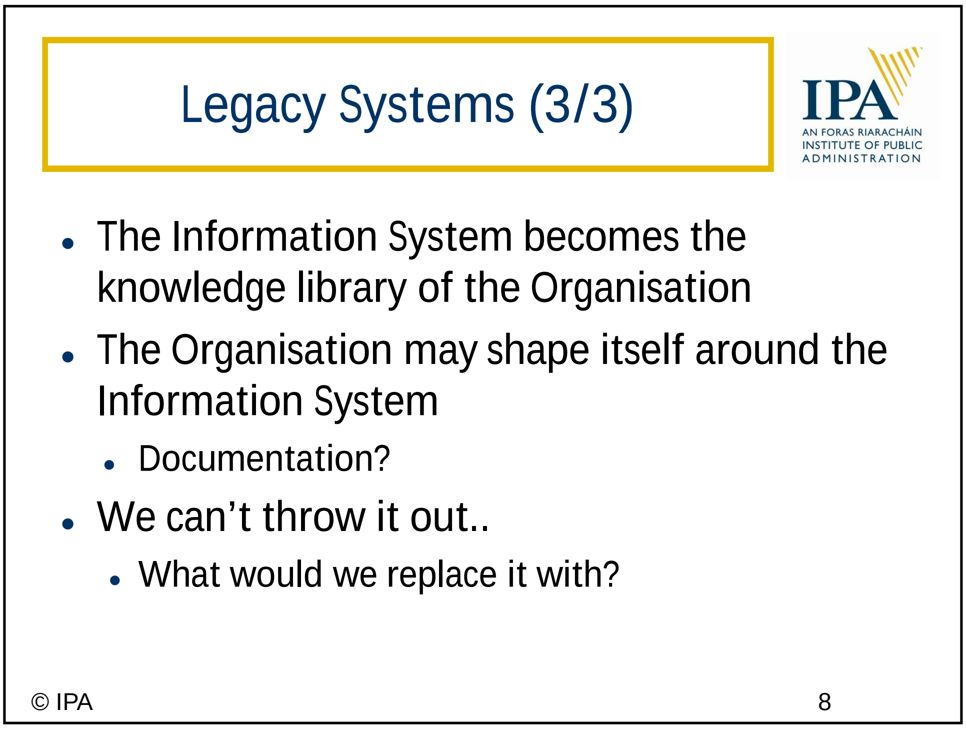
Legacy Systems (1/3)

- Many Information Systems were built when Organisations were “information islands”
- The world has moved on around them
  - They can't keep up!
  - Hard/expensive to change
  - In particular, never built to communicate with other systems



Legacy Systems (2/3)

- Years of Business knowledge in the code of the system
  - Full of edge cases, exceptions, quirks
  - Documentation is rarely adequate
  - Humans who knew that detail have often left the organisation
    - We don't always know what it's doing
    - Often, we don't know why it's being done
  - Certainly, few if any humans know all of it



Legacy Systems (3/3)

- The Information System becomes the knowledge library of the Organisation
- The Organisation may shape itself around the Information System
  - Documentation?
- We can't throw it out..
  - What would we replace it with?

Costs:

- Older Systems were built when:
  - Computers were expensive, labour was relatively cheap
  - Now, computers are cheap, and labour relatively expensive
- In a large computer system, most significant cost is now labour:
  - Must make best use of money spent on labour
  - Hardware / Operational Costs no longer the most significant!
  - Skills on older systems becoming rarer – and so generally more expensive

Increased rate of change

- Organisations have always had to change
- More external links lead to more changes
- Drivers
  - Compliance
  - Regulations
  - Market Forces
  - Globalisation
- More change, faster, and over which the Organisation has less control

Increased interconnection:

- Many Business and IT Systems were built for use within an Organisation
- What Organisations do has changed in the last 20 years:
  - Outsourcing
  - Re-selling
  - Compliance and Reporting
  - Information Systems have to support this

© IPA

Each connection unique

- Connections between existing systems are hard to make
  - Particularly older systems
- Different: platforms, languages, data definitions, plugs on the cables, operating models
- Each connection had to be "hand built"
- Tended to use "point-to-point" leased line communications

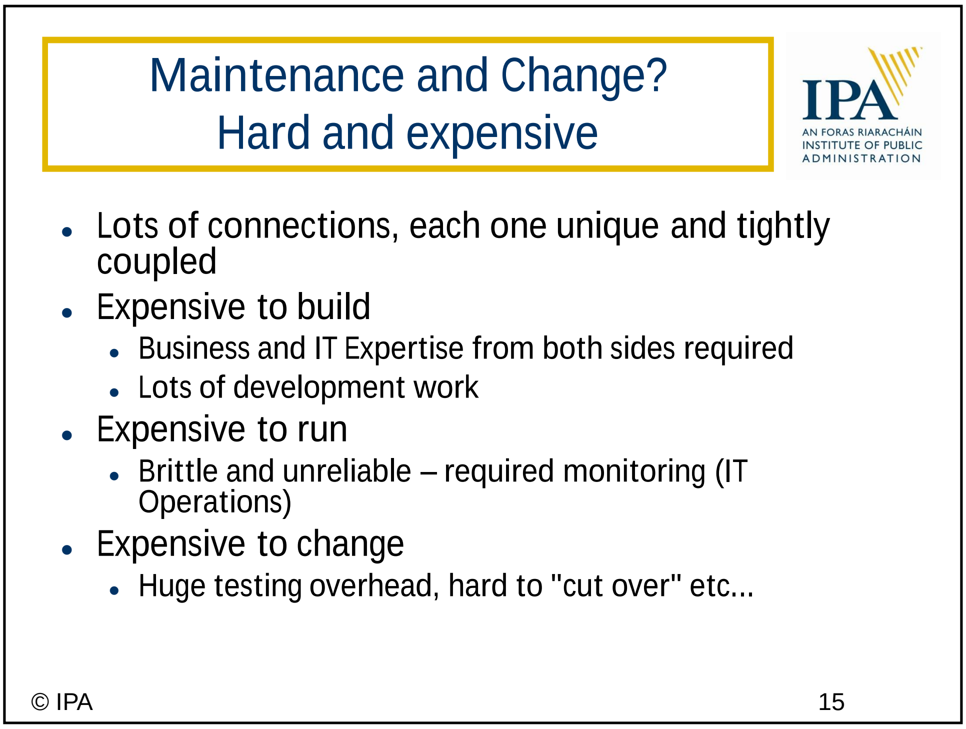
© IPA

“Spider’s Web” of connections

- More systems talking means exponential growth in connections:
  - 2 systems that need to talk – 1 connection
  - 3 systems that need to talk – 3 connection
  - 5 systems that need to talk – 16 connections!
  - etc.
- Adding the nth system adds n-1 new connections
  - This can't scale!

Tightly Coupled Connections:

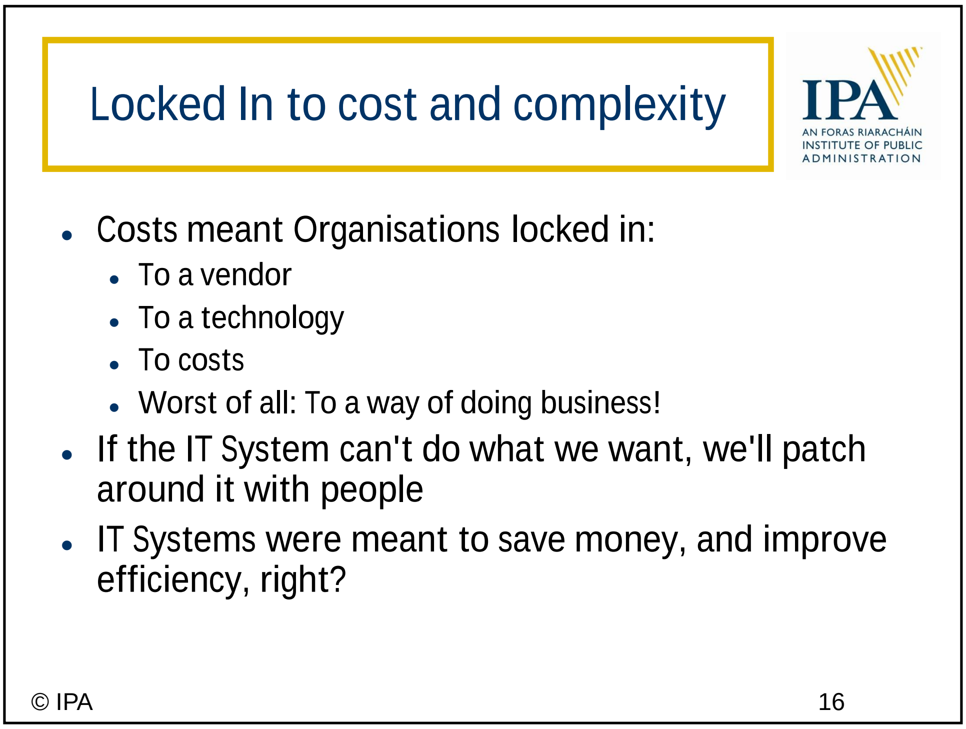
- Each side of the connection needed to know:
  - What the other side does
  - How it does those things
  - How it structures it data
  - How it fails
  - When it is switched off
  - etc...
- So when one side changes, the other had to change too!
- "Tightly Coupled"



Maintenance and Change?
Hard and expensive

- Lots of connections, each one unique and tightly coupled
- Expensive to build
  - Business and IT Expertise from both sides required
  - Lots of development work
- Expensive to run
  - Brittle and unreliable – required monitoring (IT Operations)
- Expensive to change
  - Huge testing overhead, hard to "cut over" etc...

© IPA



Locked In to cost and complexity

- Costs meant Organisations locked in:
  - To a vendor
  - To a technology
  - To costs
  - Worst of all: To a way of doing business!
- If the IT System can't do what we want, we'll patch around it with people
- IT Systems were meant to save money, and improve efficiency, right?

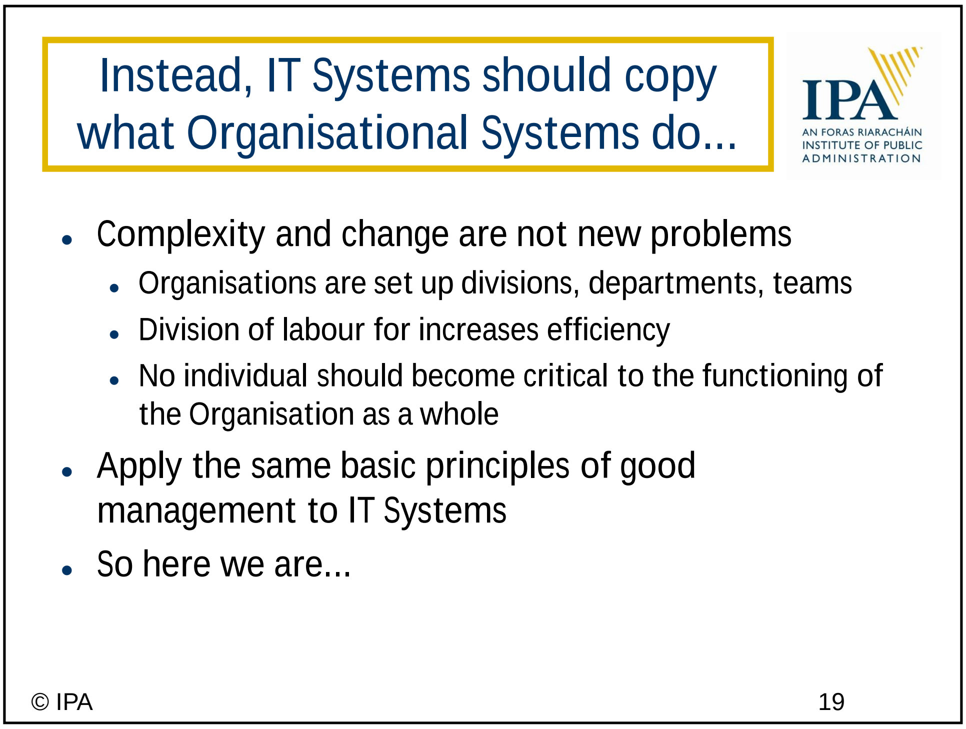
© IPA

Low re-use

- Things were built, but were hard to change or talk to from outside
  - Need significant change?
    - Might be easier to start again
  - Investment in IT became a sunk cost, and a commitment to on-going expense
  - Re-inventing the wheel every time
    - But slightly differently each time
  - Very difficult to ever shut anything down, or entirely replace it
    - Complexity and cost increases over time

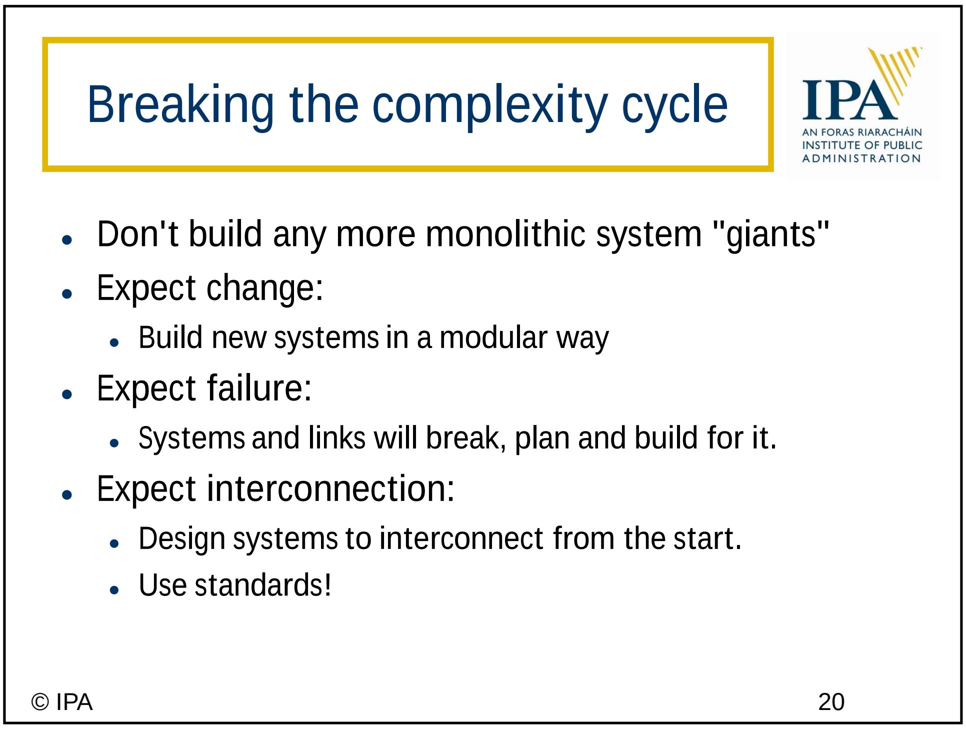
Overview

- Problems with Information Systems in practice
- **Introduction to SOA**
- Introduction to WebServices
- How SOA and WebServices relate to each other



Instead, IT Systems should copy what Organisational Systems do...

- Complexity and change are not new problems
  - Organisations are set up divisions, departments, teams
  - Division of labour for increases efficiency
  - No individual should become critical to the functioning of the Organisation as a whole
- Apply the same basic principles of good management to IT Systems
- So here we are...



Breaking the complexity cycle

- Don't build any more monolithic system "giants"
- Expect change:
  - Build new systems in a modular way
- Expect failure:
  - Systems and links will break, plan and build for it.
- Expect interconnection:
  - Design systems to interconnect from the start.
  - Use standards!

Service Oriented Architecture

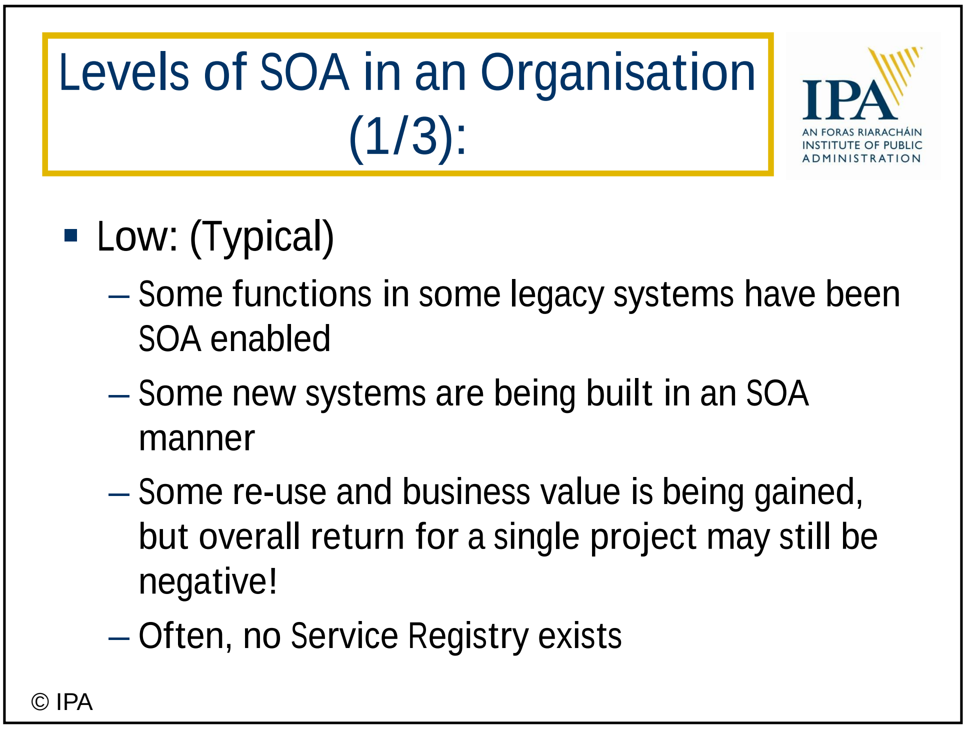
- A way of designing systems.
- Away from the problems of the past …
- Design systems to be:
  - Modular
  - Interoperable
  - Re-usable
  - Flexible

© IPA

Main Players and Operations in SOA:

- Service Provider
  - Provides Service, and publishes Service Information in Registry.
- Service Registry
  - Maintains "Yellow Pages" of services, allows requesters to find suitable services.
- Service Requester
  - Searches Registry for suitable service, binds to (uses) the service.

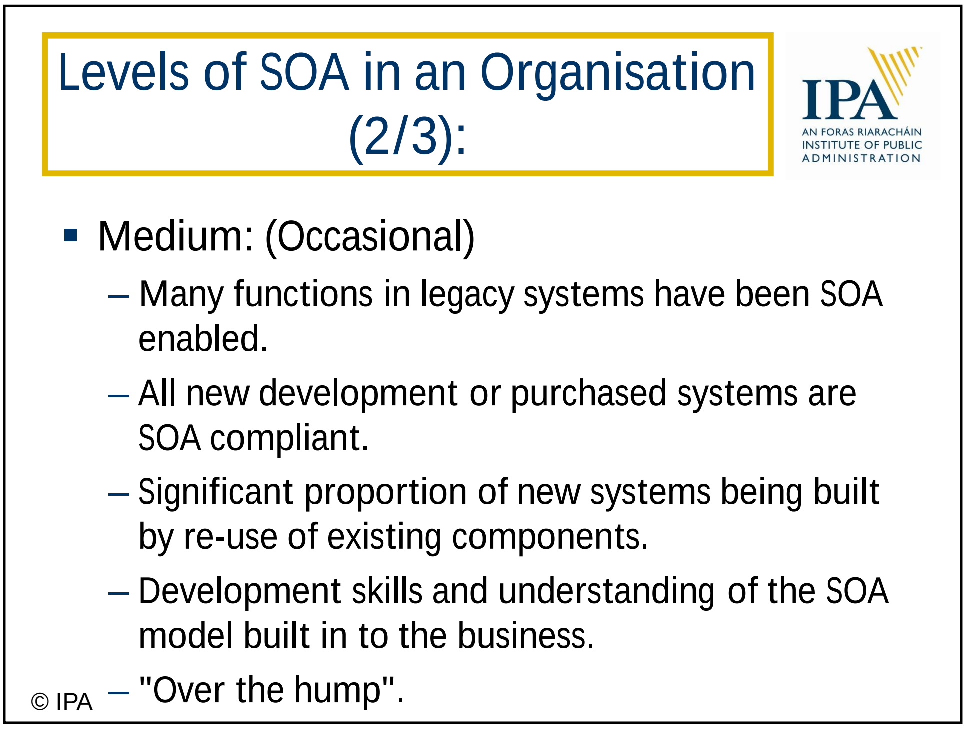
© IPA



Levels of SOA in an Organisation (1/3):

- Low: (Typical)
  - Some functions in some legacy systems have been SOA enabled
  - Some new systems are being built in an SOA manner
  - Some re-use and business value is being gained, but overall return for a single project may still be negative!
  - Often, no Service Registry exists

© IPA



Levels of SOA in an Organisation (2/3):

- Medium: (Occasional)
  - Many functions in legacy systems have been SOA enabled.
  - All new development or purchased systems are SOA compliant.
  - Significant proportion of new systems being built by re-use of existing components.
  - Development skills and understanding of the SOA model built in to the business.
  - "Over the hump".

© IPA

Levels of SOA in an Organisation (3a/3):

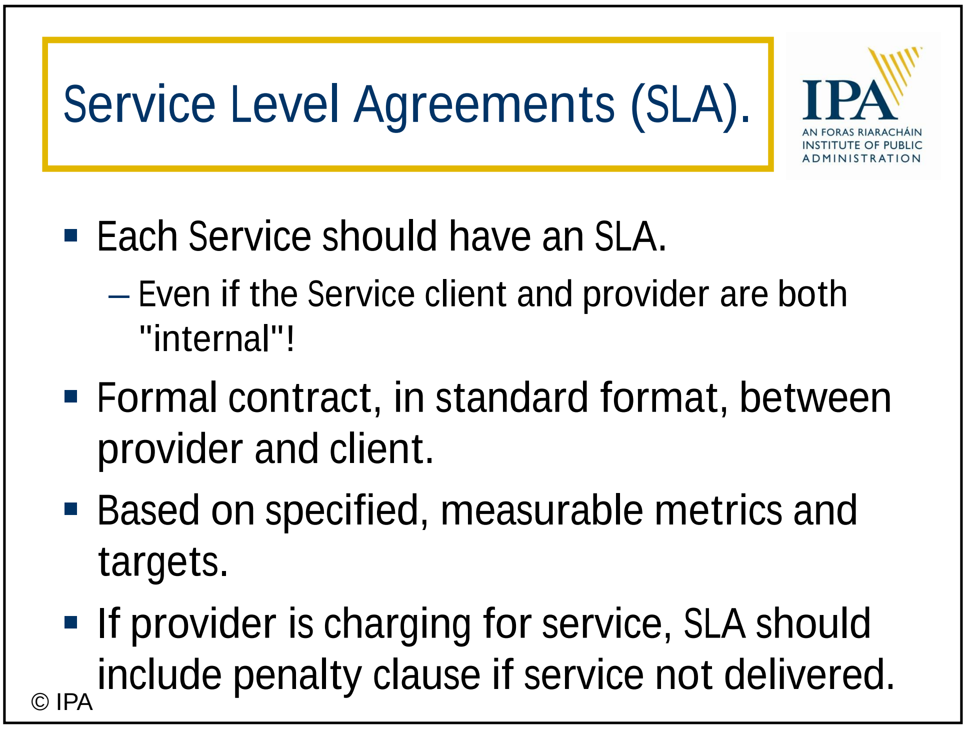
- High: (Very Rare)
  - All functions in legacy systems have been SOA enabled.
  - All new development or purchased systems are SOA compliant.
  - New development follows the SOA route, even if there’s a non-SOA existing route.
  - Only rarely is new code written, and then only in the areas of User Interface. Service Orchestration, re-use is the norm

© IPA

Levels of SOA in an Organisation (3b/3):

- High: (continued)
  - Full Service Registry and Governance processes in place.
  - Business change and opportunity driven by SOA
  - Cost of change falling
  - Flying pigs.

© IPA

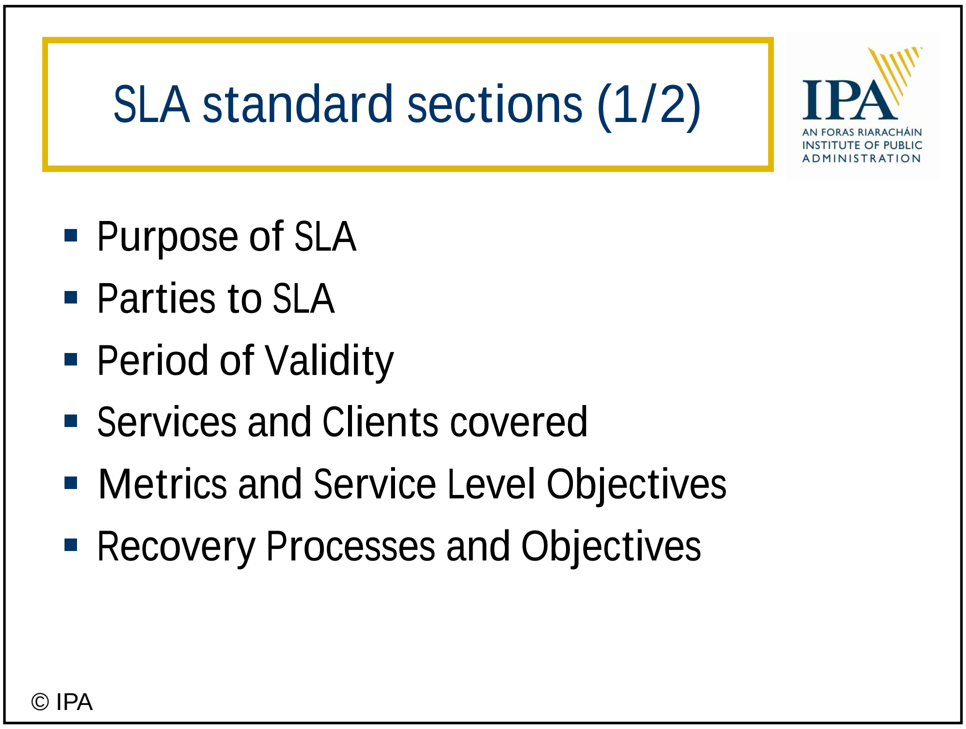


Service Level Agreements (SLA).

IPA
AN FORAS RIARACHÁIN
INSTITUTE OF PUBLIC
ADMINISTRATION

- Each Service should have an SLA.
  - Even if the Service client and provider are both "internal"!
- Formal contract, in standard format, between provider and client.
- Based on specified, measurable metrics and targets.
- If provider is charging for service, SLA should include penalty clause if service not delivered.

© IPA



SLA standard sections (1/2)

IPA
AN FORAS RIARACHÁIN
INSTITUTE OF PUBLIC
ADMINISTRATION

- Purpose of SLA
- Parties to SLA
- Period of Validity
- Services and Clients covered
- Metrics and Service Level Objectives
- Recovery Processes and Objectives

© IPA

SLA standard sections (2/2)

- Penalties
- Exclusions
- Responsibilities

© IPA

SLA Health Warnings!

- What you reward is what you'll get, for better or worse
- In an SOA architecture, easy to see what failed and when
  - Office Politics and "hot potato"
- Difficult to write a good SLA
- SLA should be a tool to ensure good service
  - More often than not, used as a weapon

© IPA

SOA Health Warnings!

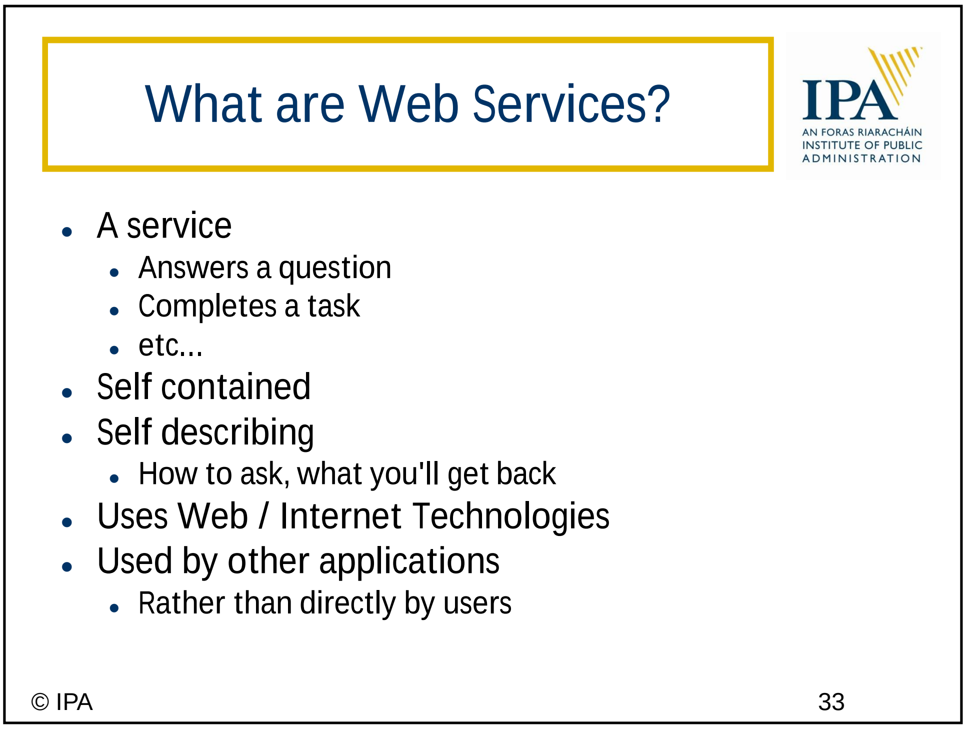
- Can't do it from the manuals and RFC docs
  - Requires experience
  - Requires getting it wrong a few times
- There's a steep starting ramp
  - Motivation, technology cost and skills
- There's significant cost before benefit
- Easier to do wrong than right
  - ..but if done right, the RoI is very real

© IPA

Overview

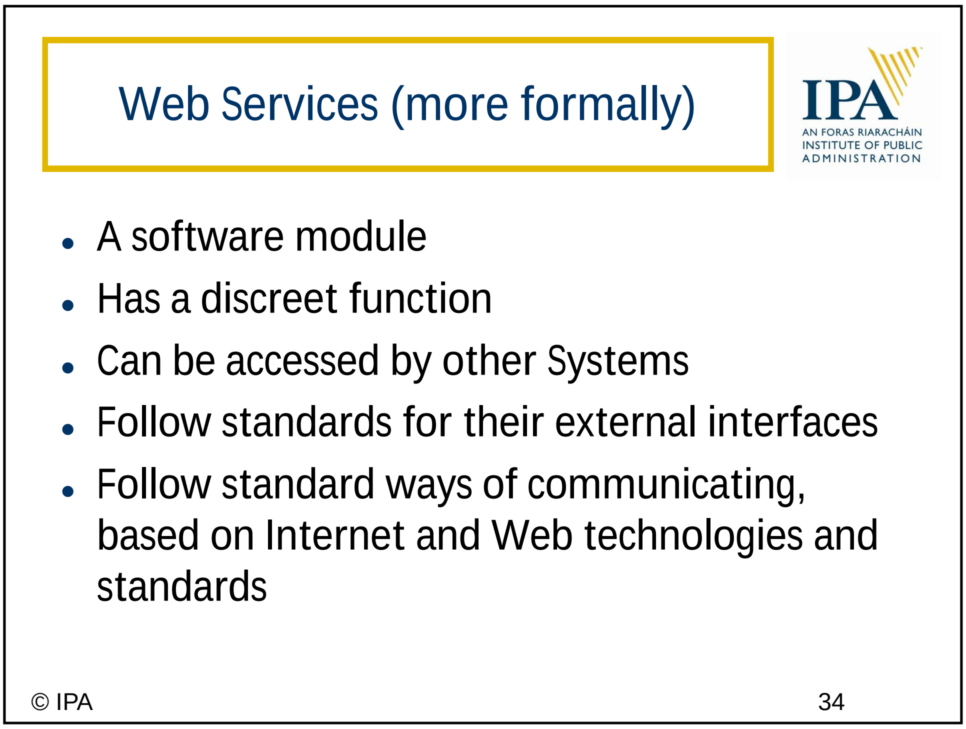
- Problems with Information Systems in practice
- Introduction to SOA
- **Introduction to WebServices**
- How SOA and WebServices relate to each other

© IPA



What are Web Services?

- A service
  - Answers a question
  - Completes a task
  - etc...
- Self contained
- Self describing
  - How to ask, what you'll get back
- Uses Web / Internet Technologies
- Used by other applications
  - Rather than directly by users



Web Services (more formally)

- A software module
- Has a discreet function
- Can be accessed by other Systems
- Follow standards for their external interfaces
- Follow standard ways of communicating, based on Internet and Web technologies and standards

"Web Services" vs. "Services over the Web".

- Web Services are not:
  - Software as a service (SAAS)
  - Directly used by humans
  - Websites
  - "The Cloud" / Platform as a Service (PAAS)
- ...though Web Services often exist behind the scenes of all of the above

Functional Properties of Web Services.

- What do I do?
  - NB: Not "How do I do it?" - you don't care how I do it.
- Where can you find me?
  - Address
- How do you ask me?
  - Format and Syntax of query
- What will you get back?
  - Format and Syntax of response
  - ...or Format and Syntax of errors

3 related concepts: (1/3)
Simple vs. Complex

- Simple:
  - Generally, answer a question
  - Request, response, finished
  - Quick, short lived
- Complex:
  - Generally mirror a business process
  - Multiple steps
  - "Answer" not immediately available
  - Generally live for longer than simple Services

© IPA

3 related concepts: (2/3)
Stateful vs. Stateless

- Do I remember anything about the last time I was called?
  - Simple informational services are generally stateless
    - What time is it?
    - How has the share price for Google changed since Wednesday?
  - Complex, transactional services generally need to be stateful
    - How long is it since I last asked you the time?
    - Has my order for shares been completed yet?

© IPA

3 related concepts: (3/3) Synchronous vs. Asynchronous.

- Synchronous:
  - Call once, get answer, done
- Asynchronous:
  - Call once, get acknowledgement of request
  - Final answer (if there is one) will come later

Loosely Coupled

- Interface:
  - Web Service interfaces (in and out) are tightly defined
- Implementation:
  - Internals are a "black box" – user of the service generally doesn't know and doesn't care how I do what I do
  - Can change anything I like internally, as long as interfaces don't change
    - Including replacing the whole thing!

Extension of "Object Oriented" programming.

- This change mirrors the change from Procedural to Object Oriented Programming
- Dominant form in new development since mid 1990's
- Relevant Principles from Object Orientation:
  - Discreet units of functionality (Class / Objects)
  - Well defined interfaces (Methods)
  - Hidden internals – the “Black Box”
- Web Services solve the same problems, but on a different scale

Granularity:

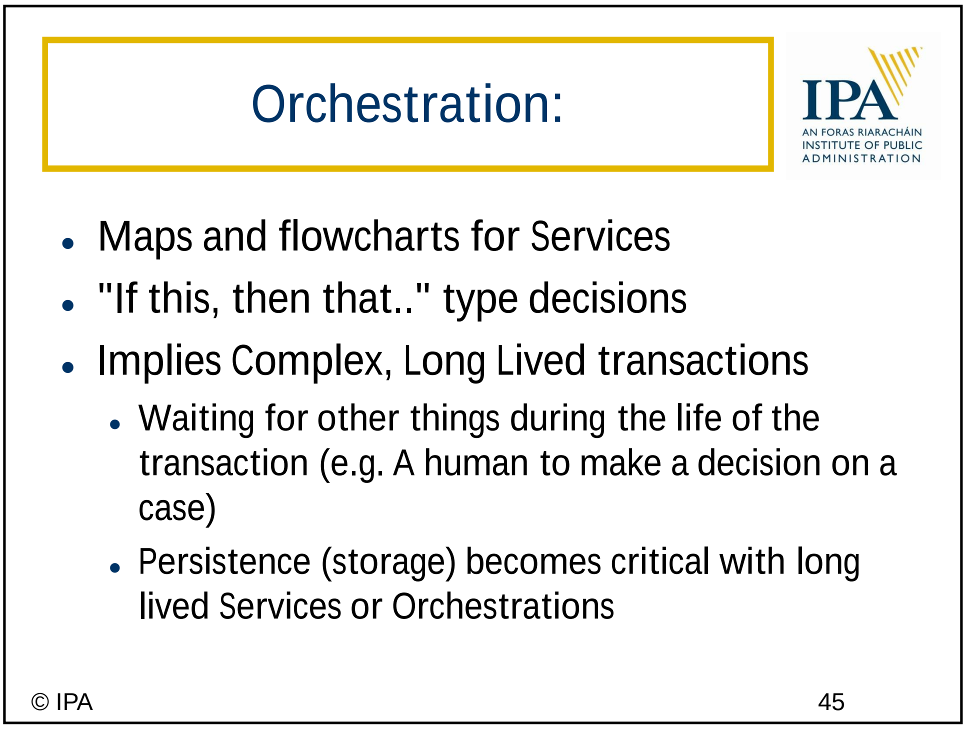
- How much should a service do?
  - Too much:
    - Hard to change
    - Hard to understand
    - Monolithic again!
  - Too little:
    - Too many tiny services
    - Hard to manage or keep track of
    - Performance?
  - "Size of function" is a crucial question.
    - No hard answer, but often goes wrong

Stackable, re-usable

- A.K.A. "Composable" or "Composite Services"
- Build simple services
- Link simple services together to provide complex services
- Expose the complex service as a web service
- Any service can be used many times, in many different contexts
- ...but again, note performance concerns!

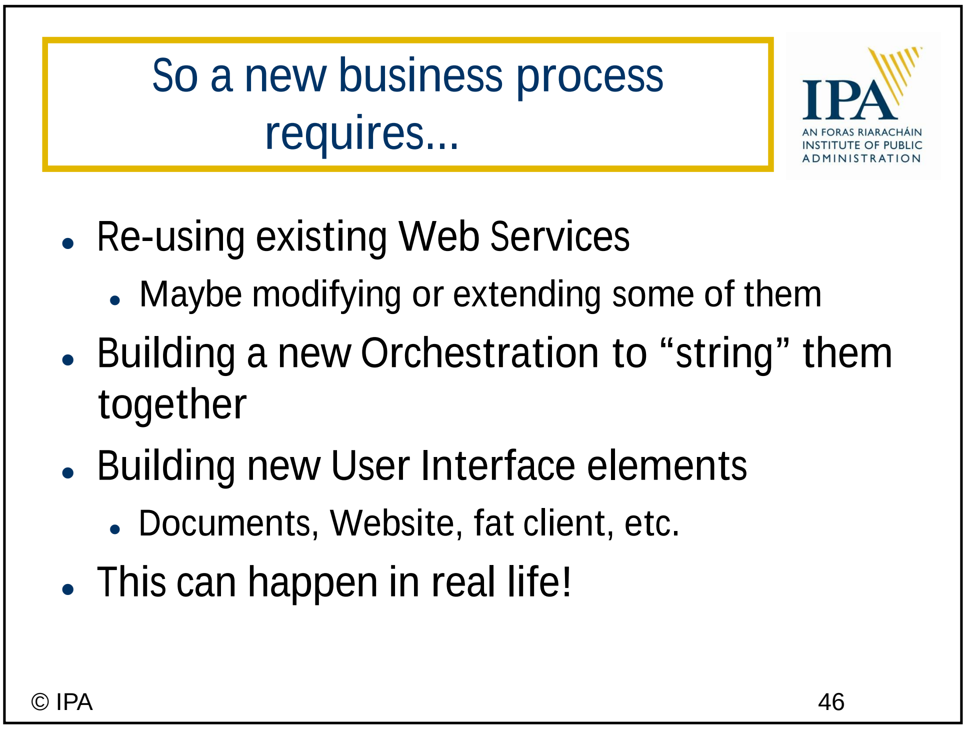
The Actors in WebServices

- Primary Actors:
  - Service Providers
  - Service Clients
  - Service Aggregators
- Mirror of the actors in SOA



Orchestration:

- Maps and flowcharts for Services
- "If this, then that.." type decisions
- Implies Complex, Long Lived transactions
  - Waiting for other things during the life of the transaction (e.g. A human to make a decision on a case)
  - Persistence (storage) becomes critical with long lived Services or Orchestrations



So a new business process requires...

- Re-using existing Web Services
  - Maybe modifying or extending some of them
- Building a new Orchestration to “string” them together
- Building new User Interface elements
  - Documents, Website, fat client, etc.
- This can happen in real life!

Non-Functional Properties of Web Services

- Response Time / Latency
- Accuracy
- Security
  - Authentication of all parties
  - Authorisation
  - Non-repudiation
- ACID
  - (atomicity, consistency, isolation, durability)
  - Difficult in a distributed environment
- ...and many more.

Quality of Service (QoS) Properties of Web Services. (1/2)

- Availability / Accessibility
  - MTBF, TTR, Sigma Rating
  - Reliability specifically under high load
- Conformance to Standards
- Conformance to Commitments (including SLA)
- Performance
- Reliability
  - Normally taken to mean "performance on a bad day".

Quality of Service (QoS) Properties of Web Services. (2/2)

- Scalability
  - Performance under abnormal load
- Security
  - Particularly where invoked over an open network
- Transactionality
  - Again, conformance to ACID principals

© IPA

WebServices are not a magic bullet!

- Easy to build, hard to build well
  - Lots to think about, lots to do, lots of (too many?) ways to do things
  - Messaging Overhead
  - Latency
  - Immature
    - Not all problems cracked yet

© IPA

Overview

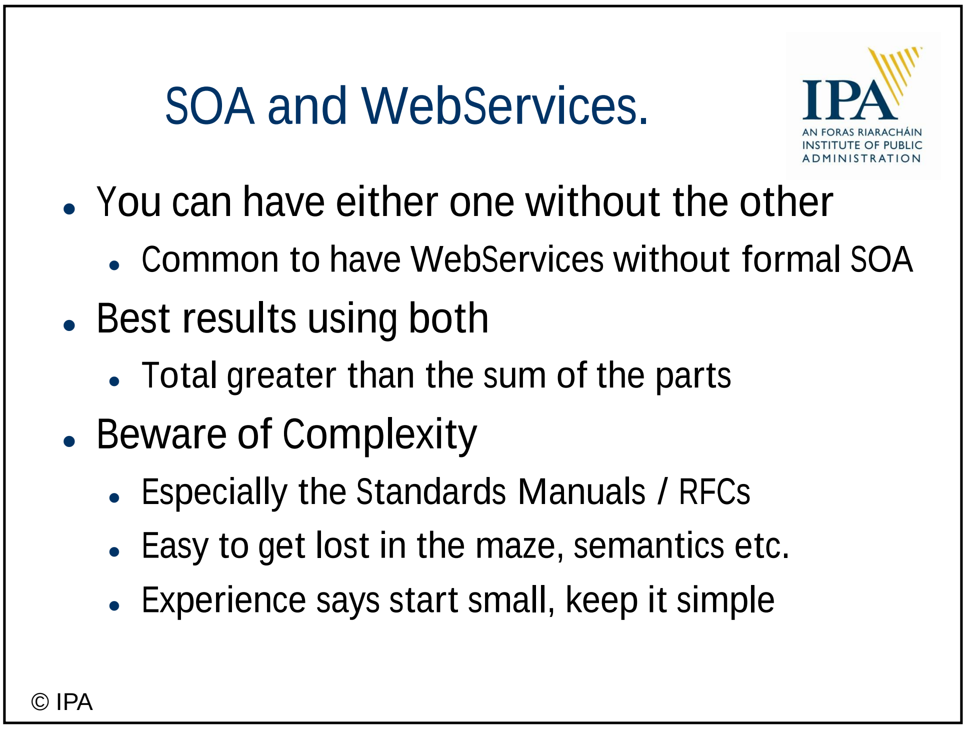
- Problems with Information Systems in practice
- Introduction to SOA
- Introduction to WebServices
- **How SOA and WebServices relate to each other**

© IPA

How do WebServices relate to SOA?

- SOA is a model of how to design IT Systems
- Web Services are a set of technologies and standards that can be used to connect Services and Systems
- SOA and Web Services fit together very well
  - There are many valid alternatives to Web Services when implementing a SOA
  - WebServices are the dominant technology at present, but are not suitable in every case

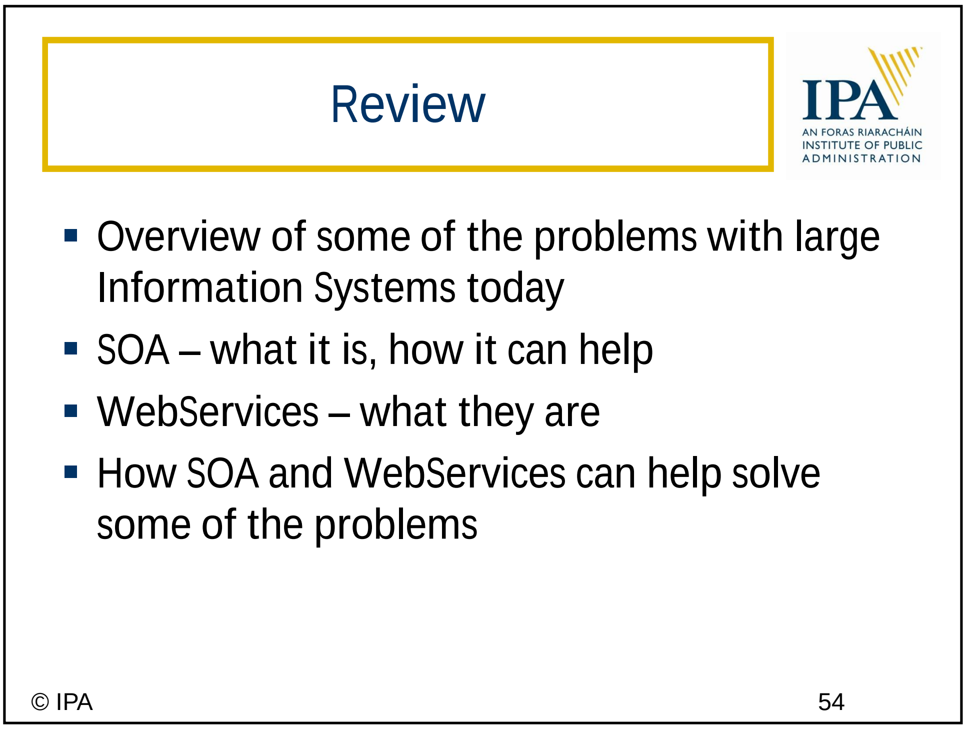
© IPA



SOA and WebServices.

- You can have either one without the other
  - Common to have WebServices without formal SOA
- Best results using both
  - Total greater than the sum of the parts
- Beware of Complexity
  - Especially the Standards Manuals / RFCs
  - Easy to get lost in the maze, semantics etc.
  - Experience says start small, keep it simple

© IPA



Review

- Overview of some of the problems with large Information Systems today
- SOA – what it is, how it can help
- WebServices – what they are
- How SOA and WebServices can help solve some of the problems

© IPA

Questions?

?