If you thought Systems Engineering was fun, you should try System of Systems Engineering!

Professor Duncan Kemp, INCOSE Fellow, CEng FIET
UK Ministry of Defence

Duncan's background

Professor Duncan Kemp, CEng FI
INCOSE Fellow

DE&S Fellow for Systems Engineering
Internal Technical Support Team Leader
Engineering Group
Abbey Wood South, BS34 8JH
Tel: +44 (0)7966 146 724

Defence Equipment & Support

1984 – Joined M

2016 – Internal Technical Support TL
2017 – Technical Discipline Lead for SE
2019 – Senior Fellow for SE
2022 – Digital Engineering Implementation TL

And …

- Chair of the INCOSE System Safety working group
- Published over 20 peer reviewed technical papers, including several on System Safety
- Presented DE&S Maritime Safety Refresher, MOD 1* Boot camp
- Guest lectured at Birmingham, Loughborough and Bristol Universities, MIT, USMA West Point
- Visiting Professor for Systems Thinking at Loughborough University
- INCOSE Fellow

Presentation overview

- Recap on Systems and Systems Engineering
- What is a System of Systems
- System of Systems Engineering Approaches
- Rail and Defence examples
- Summary and conclusions

System of Systems Engineering is …

… a way of confusing my stakeholders so I can do my job

… applying Systems Thinking beyond the system of interest

… a new name for SE

… integrating existing Systems to deliver value

… the new hype to sell consultancy and tools

… ensuring systems are planned, and specified, as part of a wider whole

… a new name for *proper* SE

… the reason for more modelling

… integrating my System with other Systems to deliver greater value

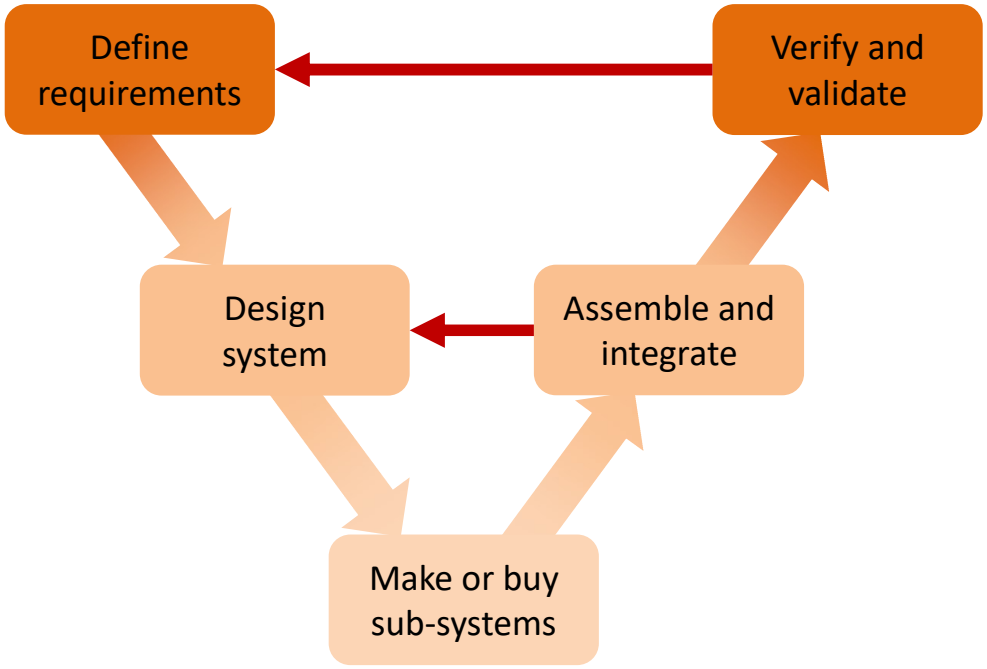
One approach to rule them all …

… integrating existing Systems to deliver value

… ensuring systems are planned, and specified, as part of a wider whole

… integrating my System with other Systems to deliver greater value

Systems refresher



System of Systems

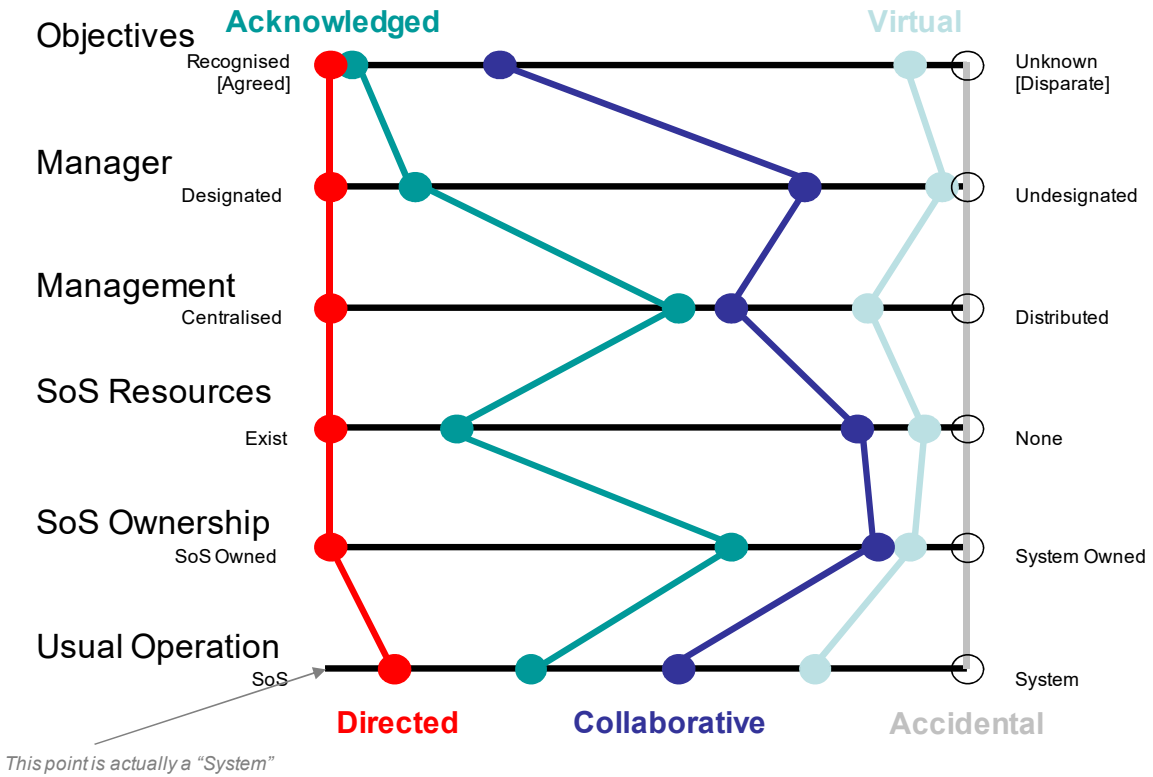
Maier's characteristics

- Operational independence
- Managerial independence
- Geographical distribution
- Emergent behaviour
- Evolutionary development

An SoS is a system, with
- Performance
- Effectiveness
- Environment
- Emergence
- Architecture
- Hierarchy
- Interfaces
- Lifecycle

But, each sub-system is also a system with its own
- Lifecycle
- Operational control
- Management control

What types of SoS are there?



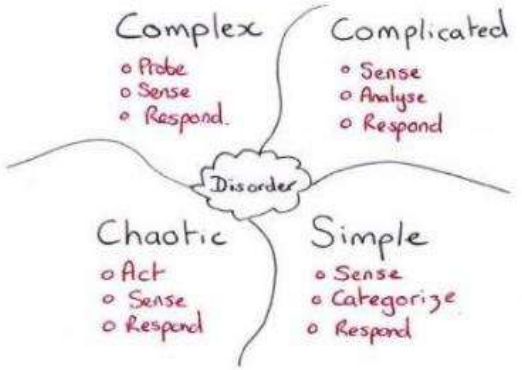
System of Systems pain points

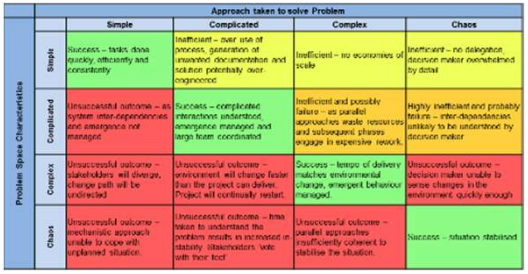
1. Authority
2. Constituent systems
3. Capabilities & Requirements
4. Autonomy, Interdependencies & Emergence
5. Testing, Validation & Learning
6. Leadership
7. SoS Principles

How to cope with no one 'in charge'

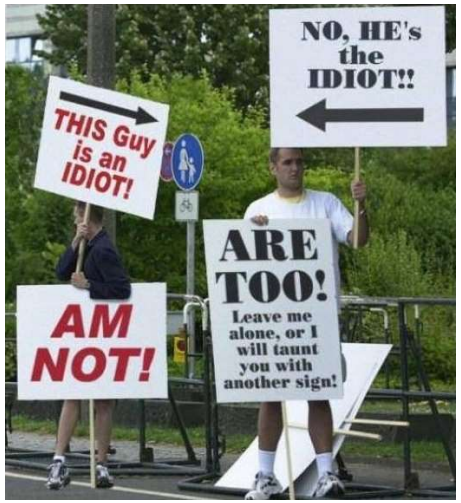
How to cope with a large complex system with 'uncontrolled' changes

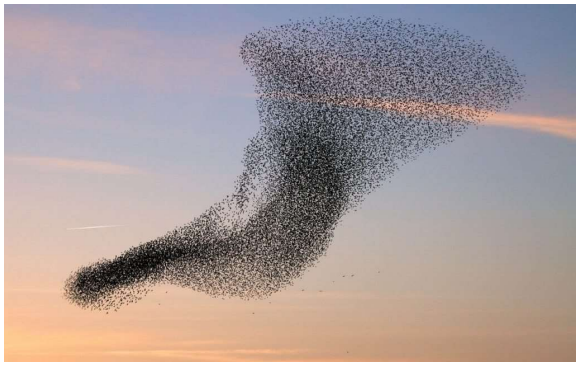
How to lead and manage in this environment



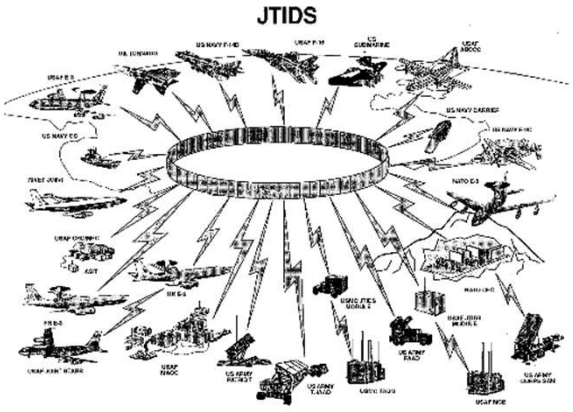


But, beware the emperors new clothes





Air Defence in the 1990s

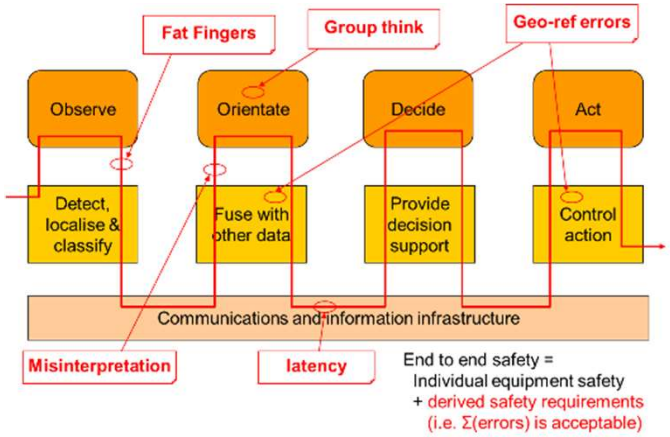


- What did we do
  - Developed an Integration architecture
  - Align operational, functional, performance and commercial perspectives
- What did we learn
  - Confusion between internal and interface standards
  - Confusion between technology, systems and operational issues
  - Functions are easy, performance is hard
  - Control is critical and difficult to get

Warship support Enterprise integration in early 2000s

- What did we do
  - Delivered a working cross organizational SoS
  - Developed a clear and effective integration architecture
  - Multiple threads, multiple lifecycles
- What did we learn
  - Followed Cynefin framework (before it was published!)
  - Commercial alignment is critical – everyone needs to think win-win
  - Operational alignment drives requirement for commercial and technical integration

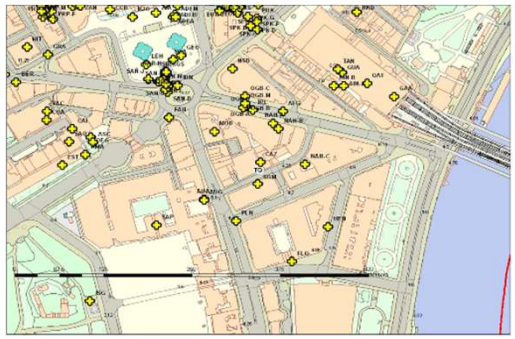
Safety critical system of systems in the mid 2000s



Variable configuration

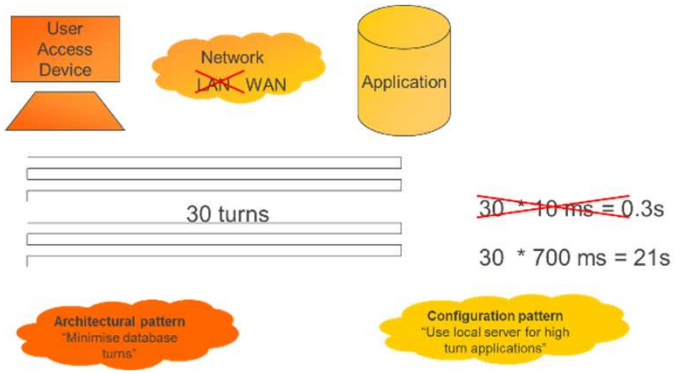
Multiple configurations

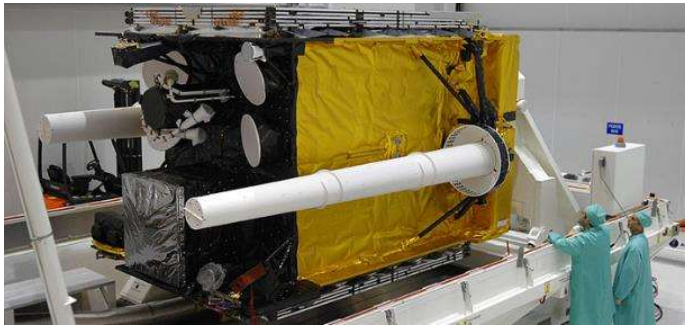
Single known configuration



- What did we do
  - Identified SoS hazards
  - Developed SoS safety approach and applied it
  - Mitigated hazards
- What did we learn
  - Safety is the ultimate emergent behavior
  - Network of safe systems can be unsafe
  - Performance can be modelled as end to end performance threads
  - Needed to constrain operational freedoms to deliver safety

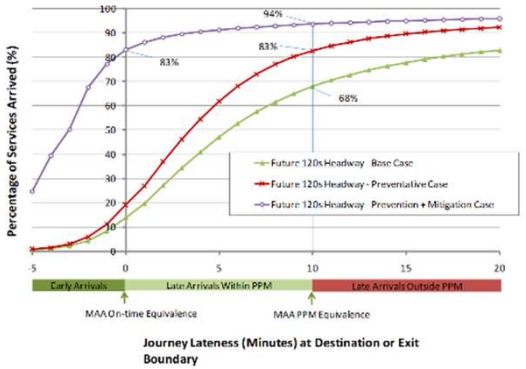
C4 integration and operations in the late 2000s





- What did we do
  - Designed and delivered end-to-end information services
  - Developed goal architecture for deployed services
- What did we learn
  - Sufficient understanding is good enough
  - Misalignment of *some* internal standards critical
  - Service engineering – Service design and Systems Engineering
  - Split the design and problem management teams
  - Synchronizing integration is so complex it is simple

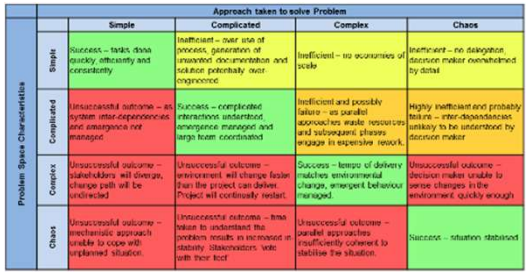
Whole systems integration in Rail in the 2010s (and 1850s)

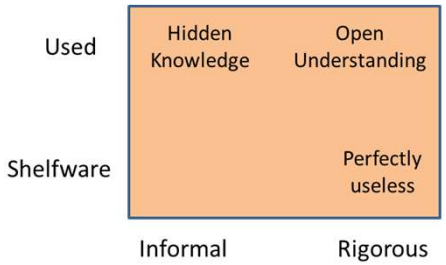


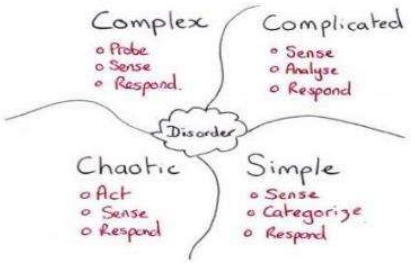
- What did we do
  - Applied defence learning to rail – and discovered it worked
  - Developed and delivered clear and successful System Integration
  - Used whole system modelling to identify significant improvements
- What did we learn
  - SoS are not as new as we think
  - There are common patterns in SoS delivery across Defence, Rail and information service delivery
  - Getting commercial incentives right is critical
  - Reality and mindsets not the same
  - Control is not as effective as influence

Accidental **Virtual** **Collaborative** **Acknowledged** **Directed** **Collaborative**

… and the lessons from the less successful







- Wrong approach for the situation
  - Planning the unplannable
  - Trying to single thread development
  - Too much rigour
  - Too little rigour
- Mindsets and processes
  - Perfection over pragmatism
  - Rigour over utility
- Modelling mayhem
  - One model to rule them all
  - Reuse before usefulness
- Aiming for the impossible
  - Field of dreams
  - Homogeneous integration

A framework for effective SoS delivery

Planning the projects to deliver,
including their role in the SoS

A framework for effective SoS delivery

Delivering the projects in the most effective way

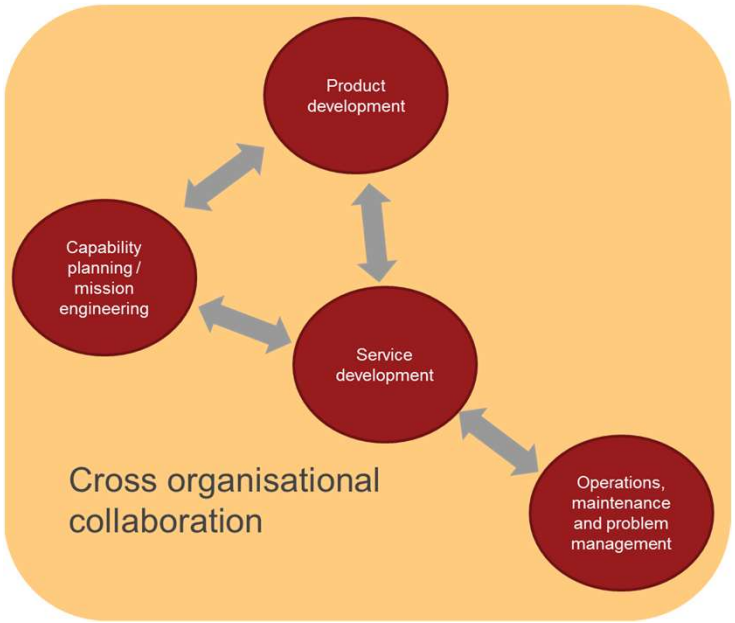
A framework for effective SoS delivery

Managing the entry and exit of systems in to service

A framework for effective SoS delivery

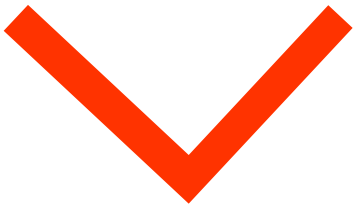
And rushing to fix problems as fast as possible

A framework for effective SoS delivery



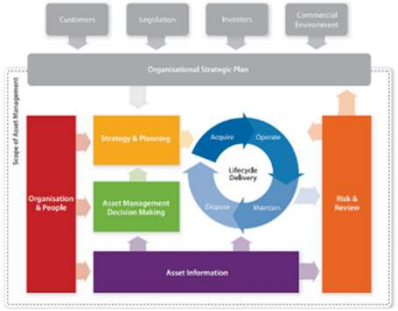
A framework for effective SoS delivery?

ISO 15288, Systems Engineering

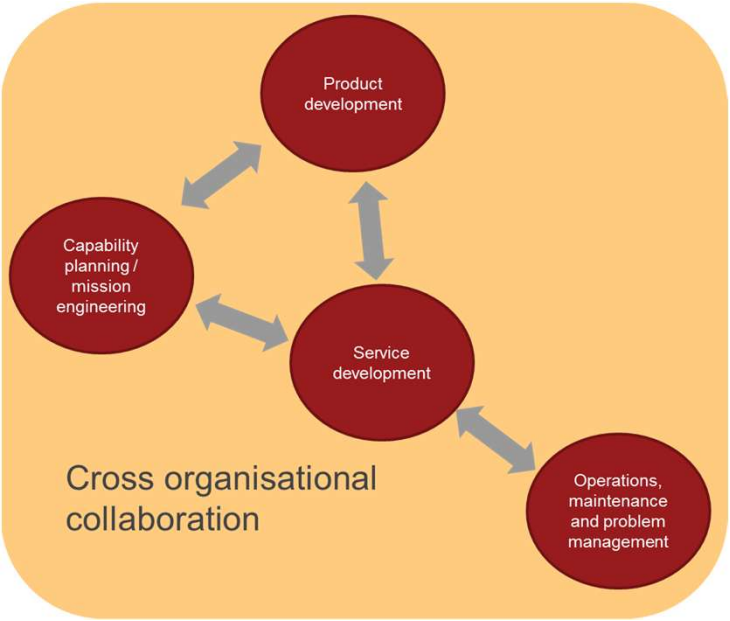


ISO 20000, Information Technology Service Management.

ISO 44001, building and sustaining collaborative business relationships



ISO 55000, optimising how the systems work together to deliver value



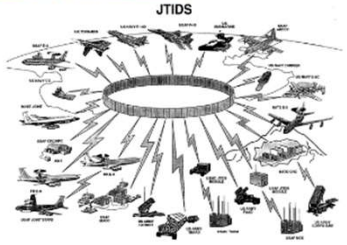
Different situations, different approaches

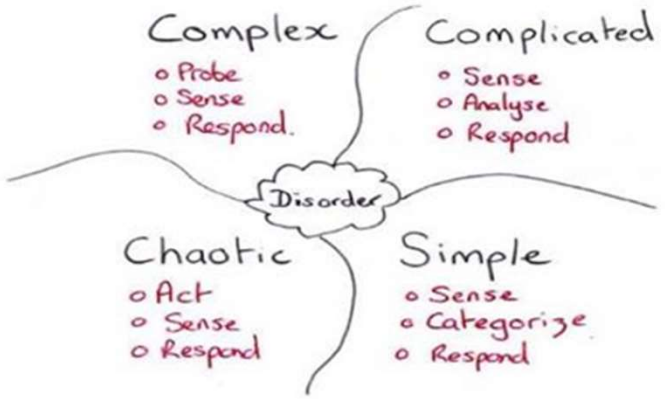
	Capability planning / mission engineering	Project and programme delivery	Service development	Operations and management
Lifecycle	Continual planning cycle	V lifecycle	Fixed time V / MoSCoW	Continuous
Outputs	Project mandates, capability architecture	Equipment, systems and products	Operational services	Service outputs and outcomes
Cynefin places	Complex and Simple	Complicated	Complicated and Complex	Simple and Chaotic
Systems standards	ISO 55000	ISO 15288	ISO 20000 and ISO 15288	ISO 20000
Management of risks	Correct in next cycle. Parallel programmes with loose coupling Emergency mid-cycle corrections	Minimise likelihood by good planning and error margins. Ask for more time and money if risks occur	Minimise likelihood by good planning and error margins. Defer deployment if risks occur	Day to day management of issues and problems Roll-back if service fails
Time horizons	My planning cycle – annual, biannual or quinquennial	As long as it takes to deliver the project – six months to fifteen years	Look out 3 to 12 months to plan my service changes	Minute by minute to day by day for management
PCTR trade approach	I will squeeze *planned* PCT as much as I can before the risk is unacceptable	I will *deliver* the performance ideally within Cost and Time. I will try to mitigate risks and make provisions if they occur.	I will deliver the greatest performance within the *absolute timescales* available. If there is a risk of delay I will revert to a guaranteed fallback.	I will trade-off *safety, capability and availability* on a minute by minute approach.

Key principals – sufficient understanding

Principle 1 – Recognise that your SoS will comprise multiple systems, working to deliver multiple services as part of a wider mission / capability. There will be multiple lifecycles and disagreements over SoS purpose and boundary.

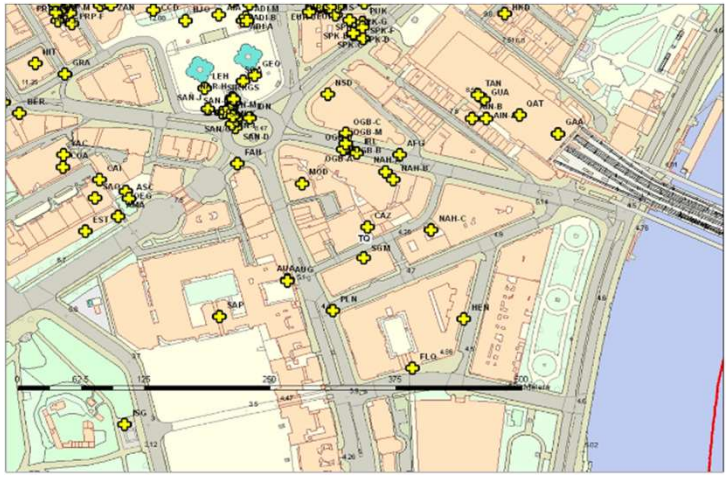
Principle 2 – Do not attempt to oversimplify the situation, focus on getting *sufficiently consistent* understanding of the Systems and SoS.

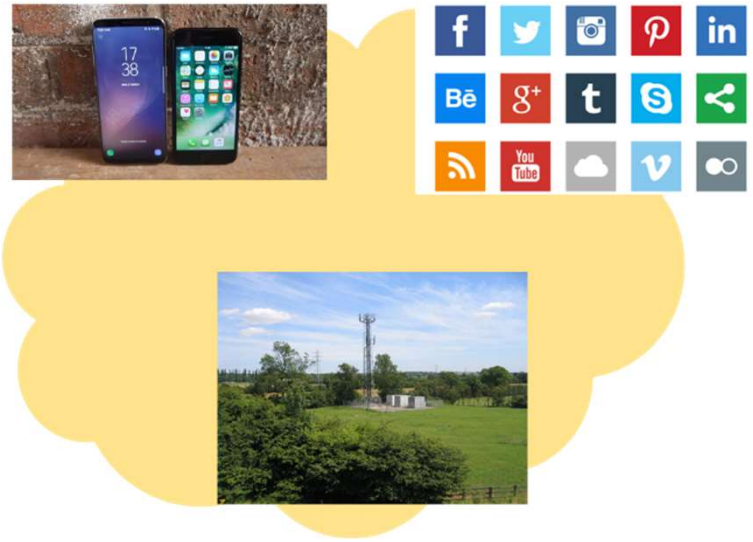




Key principals – open integration architecture

Principle 3 – Develop an integration architecture that describes the functionality, performance and commercial / organisational aspects of the SoS. The more open and modular the architecture, the easier it will be to evolve the SoS.

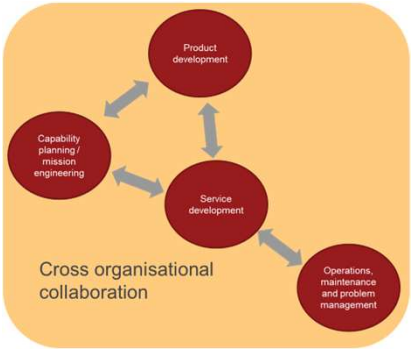




Key principals – different but aligned approaches

Principle 4 – Use 4 different approaches for the 4 different enterprises. Implement the appropriate practice for each approach. Good practice for one approach may not be good practice for another – one size does not fit all. The processes, tools, culture/mindsets, management and leadership styles need to be different

Principle 5 – Integrate the four different approaches. Remember that the are collaborating enterprises, not stages in a lifecycle. Ensure everyone has sufficient understanding of the different approaches to enable others to deliver.



Key principals – collaborate to mutual benefit

Principle 6 – Balance the costs and benefits for everyone participating in the SoS. The more participants want to be in the SoS, the more committed they will be to fix problems and expand use.

Principle 7 – Focus on the easiest and highest value services to improve. The epitome of skill is not to build an expensive new system to deliver a new service, it is to deliver the new service by reusing the existing systems.

Conclusions