Chapter 20 – Systems of Systems

Topics covered

- System complexity
- System of systems classification
- Reductionism and complex systems
- Systems of systems engineering
- Systems of systems architecture

Systems of systems

- More and more systems are being constructed by integrated existing, independent systems
- *A system of systems is a system that contains two or more independently managed elements.*
- There is no single manager for all of the parts of the system of systems and that different parts of a system are subject to different management and control policies and rules.

Examples of systems of systems

- A cloud management system that handles local private cloud management and management of servers on public clouds such as Amazon and Microsoft.
- An online banking system that handles loan requests and which connects to a credit reference system provided by credit reference agency to check the credit of applicants.
- An emergency information system that integrates information from police, ambulance, fire and coastguard services about the assets available to deal with civil emergencies such as flooding and large-scale accidents.

Essential characteristics of SoS

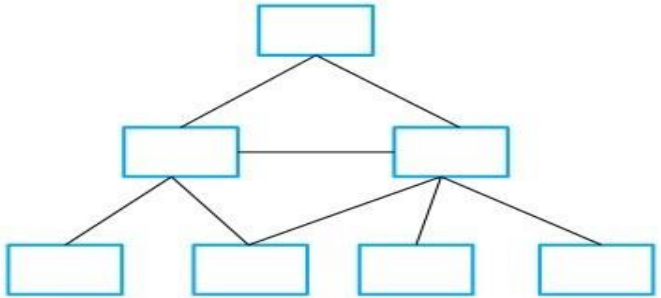
- ✧ Operational independence of system elements
- ✧ Managerial independence of system elements
- ✧ Evolutionary development
- ✧ Emergence of system characteristics
- ✧ Geographic distribution of system elements
- ✧ Data intensive (data >> code)
- ✧ Heterogeneity

System complexity

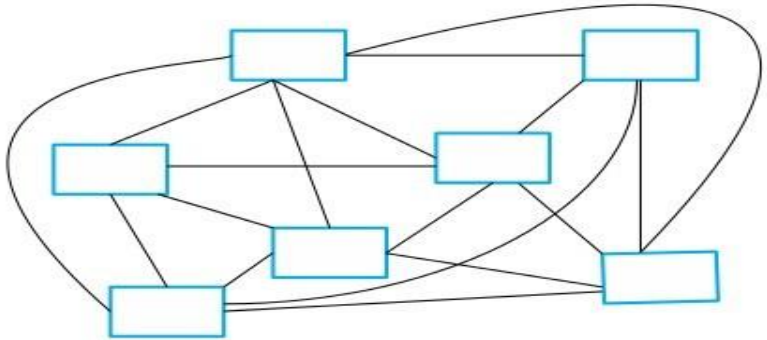
Complexity

- All systems are composed of parts (elements) with relationships between these elements of the system.
  - For example, the parts of a program may be objects and the parts of each object may be constants, variables and methods.
  - Examples of relationships include 'calls' (method A calls method B), 'inherits-from' (object X inherits the methods and attributes of object Y) and 'part of' (method A is part of object X).
- The complexity of any system depends on the number and the types of relationships between system elements.
- The type of relationship (static or dynamic) also influences the overall complexity of a system.

Simple and complex systems



System (a)

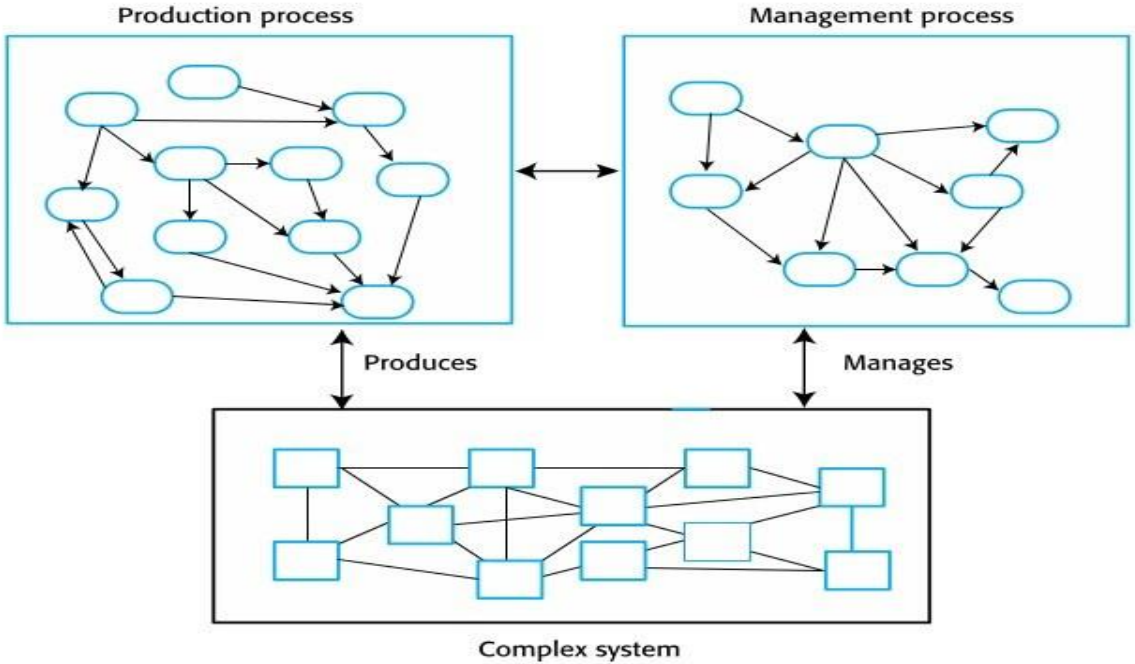


System (b)

Process complexity

- As systems grow in size, they need more complex production and management processes.
- Complex processes are themselves complex systems.
  - They are difficult to understand and may have undesirable emergent properties. They are more time consuming than simpler processes and they require more documentation and coordination between the people and the organizations involved in the system development.
- The complexity of the production process is one of the main reasons why projects go wrong, with software delivered late and over-budget.

System production and management processes



Complexity and software engineering

- Complexity is important for software engineering because it is the main influence on the understandability and the changeability of a system.
- The more complex a system, the more difficult it is to understand and analyze.
- As complexity increases, there are more and more relationships between elements of the system and an increased likelihood that changing one part of a system will have undesirable effects elsewhere.

Types of complexity

- *Technical complexity* is derived from the relationships between the different components of the system itself.
- M*anagerial complexity* is derived from the complexity of the relationships between the system and its managers and the relationships between the managers of different parts of the system.
- *Governance complexity* of a system depends on the relationships between the laws, regulations and policies that affect the system and the relationships between the decision-making processes in the organizations responsible for the system.

System characteristics and complexity

SoS characteristic	Technical complexity	Managerial complexity	Governance complexity
Operational independence		X	X
Managerial independence	X	X	
Evolutionary development	X		
Emergence	X		
Geographical distribution	X	X	X
Data-intensive	X		X
Heterogeneity	X		

Complexity and project failure

- Large-scale systems of systems are now unimaginably complex entities that cannot be understood or analyzed as a whole.
- The large number of interactions between the parts and the dynamic nature of these interactions means that conventional engineering approaches do not work well for complex systems.
- It is complexity that is the root cause of problems in projects to develop large software-intensive systems, not poor management or technical failings.

Systems of systems classification

Maier's classification of systems of systems

- **Directed SoS** are owned by a single organization and are developed by integrating systems that are also owned by that organization. The system elements may be independently managed by parts of the organization.
- **Collaborative SoS** are systems where there is no central authority to set management priorities and resolve disputes. Typically, elements of the system are owned and governed by different organizations.
- **Virtual systems** have no central governance and the participants may not agree on the overall purpose of the system. Participant systems may enter or leave the SoS.

More intuitive classification terms

- *Organizational systems of systems* are SoS where the governance and management of the system lies within the same organization or company.
- *Federated systems* are SoS where the governance of the SoS depends on a voluntary participative body in which all of the system owners are represented.
- *System of system coalitions* are SoS where there are no formal governance mechanisms but where the organizations involved informally collaborate and manage their own systems to maintain the system as a whole.

System of systems classification