Services and SOA

CMPU4023 - Enterprise Application Development

History

- In the early stages of business information systems development and data processing, enterprise information systems were highly centralised in a small number of locations in expensive minicomputers or mainframe systems, controlled by a smaller number of gatekeeper experts
- Users interfaced using primitive text terminals but reporting and analysis continued to be largely paper-based
- With the advent of the personal computer, users began to use standalone software packages to do some data processing and analysis which led to a fracturing in the cohesion of the enterprise information systems
- This problem began to be solved when local area networks began to stitch these previously siloed data repositories

Monolithic Software

- Owing largely to their centralised origins, data processing applications tended to be monolithic in nature, i.e. a small number of large applications providing many diverse functions on a wide variety of data sources
- The problem with this approach for the business and the end-user was the cost of maintaining such large and complex systems and the lack of flexibility to be able to provide new or bespoke solutions in a timely manner
- In favour now, the monolithic applications have been decomposed into smaller single purpose services which can be recomposed by consumers in some manner particular to their business needs

Service-Oriented Architectures

- In simple terms, a *service-oriented architecture* is a style of software design which sees the construction of software solutions from a set of technology-independent components which can be composed together over a network using some well-defined network protocol
- This contrasts with a monolithic application in the sense that solutions to problems not originally address by the monolith can be created by end-users more flexibly from the components
- The components that make up SOAs are known are services
- Services are the basic unit of development in a SOA environment

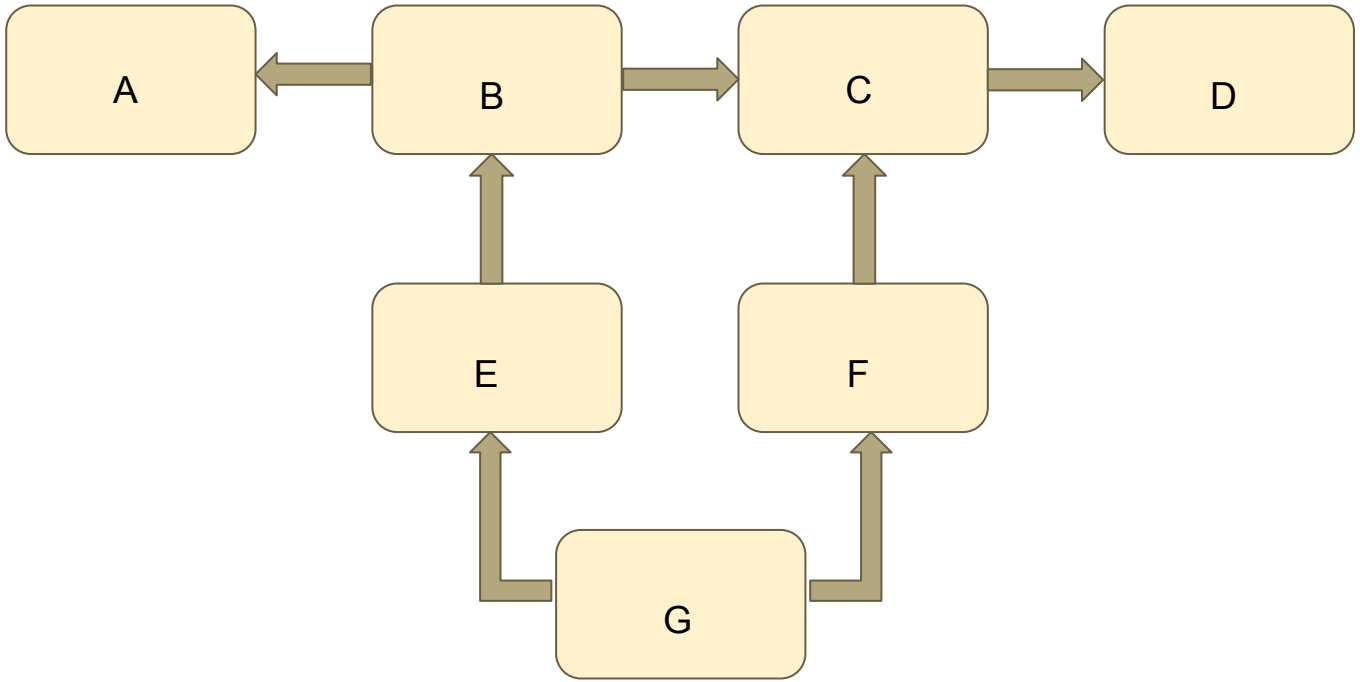
Services

- Services have four basic properties in the SOA design approach

1. Logically represents some business activity with a defined outcome
2. It completely self-contained
3. Is considered to be a black-box from a consumer's perspective
4. May itself be composed from one or more other services

- A service abstracts the details of the business logic it implements by presenting some canonical view to the consumer
- The consumer expects that service to be able to fulfill its specialised task without the need for external dependencies
- Service composition is a natural graph in nature with loose coupling

Service Composition



Microservices

- Some of the most recent literature in SOA talks about the idea of *microservices* although there is no clearly defined notion of what makes a microservice “micro”
- However, the design philosophy here is to maintain the discipline of having services remain small, focused, compact and to do one thing well and one thing only
- More complex service interfaces can then be built atop of microservices to implement and expose more nuanced business functions

SOA Implementations

- SOA is independent of any particular implementation technology - it is a style of software systems design
- That leaves a real-world enterprise implementation potentially a wide variety of technology options to choose from
- In practice, however, there are often many constraints on what is actually possible
- Implementations must consider the available software and skills and the network operating constraints and mandated policies
- Enterprise networks are very unforgiving environments to develop for, a theme that we will return to in later lectures

Web Technologies

- In recent years, enterprises have adopted much of the web technology stack for services implementation
- In particular, HTTP is the favoured application-layer protocol and XML and JSON are the favoured serialisation formats
- As a set of industry standards with wide community support, the web stack offers cost-saving solutions to common service construction concerns
- However, services using web technologies need not necessarily be consumed on the web or by web clients in the traditional sense
- Here, the web is just a reusable toolkit of common building blocks

Enterprise, Open Source and Open Standards

- Because of their increasing reliance on open standards and open source software components within their own businesses, enterprises are tending to contribute to the development and support of these in the public fora
- It is not uncommon for the larger technology companies to provide resources or financial assistance to standards bodies and software projects in the hope of both maintaining their ongoing viability and influencing their strategic and technical direction

Services as Units of Development

- In addition to being the black-box abstractions of a business function, a service is also a natural software sub-assembly from a development perspective analogous in many ways to reusable libraries
- Service development can be treated as standalone projects in their own right meaning a dedicated set of resources, management processes and scheduling for a specific, relatively narrowly-defined set of deliverables
- This decoupling and independence of project development has advantages for large teams albeit that resources and underlying libraries would be shared in many cases, reintroducing a certain level of coupling and depenedency

Testing and Reliability

- Monolithic systems are hard to test reliably because good test coverage in a large system is very hard and quite often infeasible
- In contrast, services with their relatively narrowly-defined interfaces and functions are much easier to reason about and construct a set of tests for
- When testing is so feasible, it can be automated using techniques like test-driven-development (TDD)
- When automated tests exist, these can be run as often as is required and can be used a quality gatekeepers of service deployment using techniques such as continuous integration (CI)
- Both TDD and CI will be considered in more detail in later lectures

Summary

- Monolithic software applications with their centralised origins have become less popular in enterprises in recent years
- SOAs have begun to replace the applications providing specialised, single business function services which can be composed more flexibly and quickly to solve business IS needs
- Services are easier to construct, maintain and test and offer considerable cost and time advantages
- Services are often built from open source and standards software building blocks to which many enterprises contribute resources and intellectual property rights