**The Common Statistical Production Architecture (CSPA) and the Open Group Service Integration Maturity Model (OSIMM)**

CSPA doesn’t defines a maturity model, but refers to The Open Group Service Integration Maturity Model (OSIMM), which specifies how to measure the service integration levels of an organization and its IT systems and business applications.

OSSIM has seven dimensions across seven maturity levels. Their dimensions are:

1. **Business**, is focused on the business architecture, address issues like business practices and policies; how business are designed, structured, implemented, and executed. The Business dimension also addresses how the cost of IT capabilities is allocated across the enterprise, and how well the IT capabilities support the flexibility of the business, agility, and SLAs. The Business dimension includes IT strategy.
2. **Organization and governance**, is focused on the structure and design of the organization itself and the necessary measures of organizational effectiveness in the context of an SOA and SOA governance. The Organization aspect is focused on organizational structure, relationships, roles, and the empowerment necessary to adopt a service-oriented strategy. This includes the types and extent of skills, training, and education that are available within the organization. Governance is associated with formal management processes to keep IT activities, service capabilities, and SOA solutions aligned with the needs of the business.
3. **Method**, is focused on the methods and processes employed by the organization for its IT and business transformation, and the organization’s maturity around the Software Development Lifecycle such as the use of requirements management, estimation techniques, project management, quality assurance processes, design methodologies and techniques, and tools for designing solutions.
4. **Application**, is focused on application style, structuring of the application and functional decomposition, re-usability, flexibility, reliability, and extensibility of the applications, understanding, and uniform use of best practices and patterns, whether multiple applications have been created to serve different lines of business with essentially the same functionality, and the availability of enterprise schema and object models.
5. **Architecture**, is focused on the structure of the architecture which includes topology, integration techniques, enterprise architecture decisions, standards and policies, web services adoption level, experience in SOA implementation, SOA compliance criteria, and typical artifacts produced.
6. **Information**, is focused on how information is structured, how information is modeled, the method of access to enterprise data, abstraction of the data access from the functional aspects, data characteristics, data transformation capabilities, service and process definitions, handling of identifiers, security credentials, knowledge management, business information model, and content management.
7. **Infrastructure and management**, is focused on the organization’s infrastructure capability, service management, IT operations, IT management and IT administration, how SLAs are met, how monitoring is performed, and what types of integration platforms are provided.

And the seven maturity levels are:

1. **Silo**. Individual parts of the organization are developing their own software independently, with no integration of data, processes, standards, or technologies. This severely limits the ability of the organization to implement business processes that require co-operation between the different parts, and the IT systems cannot be integrated without significant manual intervention, such as re-keying and re-interpretation of data.
2. **Integrated**. Technologies have been put in place to communicate between the silos, and to integrate the data and interconnections. The construction of an IT system that integrates across different parts of the organization becomes possible. However, integration does not extend to common standards in data or business processes. Therefore, to connect two systems, it requires a, possibly complex, conversion of the data, operations, and protocols used by these systems. Each such connection may require bespoke code and adapters, leading to a proliferation of software that is difficult to manage and complex to code. It is therefore not easy to develop or automate new business processes.
3. **Componentized**. The IT systems in the silos have been analyzed and broken down into component parts, with a framework in which they can be developed into new configurations and systems. There may also be some limited analysis of the business functionality into components. Although components interact through defined interfaces, they are not loosely coupled, which limits agility and interoperability between different segments of the organization (or even different organizations within the business “eco-system”). This causes difficulties in development and deployment of shared business processes. Business and infrastructure components are discrete and re-usable through code and EAI re-use techniques. However, they are often replicated and redundant.
4. **Service**. Composite applications are built from loosely-coupled services. The way that services may be invoked is based upon open standards and is independent of the underling application technology. Services run on an IT infrastructure that is supported by the appropriate protocols, security mechanisms, data transformation, and service management capabilities. The services may therefore interoperate across all of the parts of the organization and even across different organizations within the eco-system, and are often managed by assigning responsibilities for managing Service-Level Agreements (SLAs) to segments of the organization. The business functionality has been analyzed in detail and is broken down into services residing within a business architecture that ensures that services will interoperate at the business level. In addition, it is possible to define the services via a specification language – such as WSDL or Service Component Architecture (SCA) – that unambiguously defines the operations performed by the service, permitting the construction of a catalog of services. The combination of IT and service architectures permits the construction of systems based upon these services, operating right across the organizations in the ecosystem. However, at this stage the composition of services and flow of control within a composite application are still defined by developers writing bespoke code, rather than by a declarative flow language. This limits the agility of the development of new business processes as services.
5. **Composite Services**. At this level of service maturity it is now possible to construct a business process for a set of interacting services, not just by bespoke development, but by the use of a composition or business process modeling language, such as BPEL [BPEL] of information and control through the individual services. Composite services include static, process, and activity-based services. This permits the assembly of services into composite business processes, which may be short or long running, without significant construction of code. Thus, the design and development of services is agile, and may be performed by developers under the close guidance of business analysts.
6. **Virtualized Services**. The business and IT services are now provided through a façade – a level of indirection. The service consumer does not invoke the service directly, but through the invocation of a “virtual service”. The infrastructure performs the work of converting the virtual invocation into a physical call of the real service, and may as part of this conversion change the address, the network, the protocol, the data, and the synchronization pattern that is contained in the call. Such conversions may be a complex service in their own right, such as the transformation of data from one data model to another. The virtual service thereby becomes more loosely coupled from the infrastructure on which it is running, permitting more opportunities for the composition of services. This is in contrast to the lower levels of service maturity, where the service is more closely coupled to the infrastructure. Although virtualization has been used in non-SOA systems, this level extends the concept (and advantages) of virtualization to services.
7. **Dynamically Re-Configurable Services**. Prior to this level, the business process assembly, although agile, is performed at design time by developers (under the guidance of business analysis and product managers) using suitable tooling. Now this assembly may be performed at runtime, either assisted by the business analysts via suitable tooling, or by the system itself. This requires the ability to access a repository of services and to query this repository via the characteristics of the required services. In its simplest form, these characteristics may have been defined in advance, restricting the system to selecting and locating specific instances of services.

