V. CSPA 2.0 Application Architecture

98. TOGAF provides the following useful definitions:

- **Application** - A deployed and operational IT system that supports business functions and services; for example, a payroll. Applications use data and are supported by multiple technology components but are distinct from the technology components that support the application.
- **Architecture** - 1. A formal description of a system, or a detailed plan of the system at component level, to guide its implementation (source: ISO /IEC 42010:2007). 2. The structure of components, their inter-relationships, and the principles and guidelines governing their design and evolution over time.
- **Application Architecture** - A description of the major logical grouping of capabilities that manage the data objects necessary to process the data and support the business.
- **Technology Architecture** - The logical software and hardware capabilities that are required to support deployment of business, data, and application services. This includes IT infrastructure, middleware, networks, communications, processing, and standards.

99. CSPA uses the following definition that blends several important elements from these definitions:

"Application architecture is a description of the major logical grouping of capabilities that manage the data objects necessary to process the data and support the business - it details the structure of components, their inter-relationships, and the principles and guidelines governing their design and evolution over time."

100. As described in Section II, The CSPA Application Architecture is based on an architectural style called Service Oriented Architecture (SOA). This style focuses on Services (or CSPA Services in this case). A service is a representation of a real world business activity with a specified outcome. It is self-contained and can be reused by a number of business processes (either within or across statistical organizations).

101. CSPA Services have interfaces that are called to perform business processes. SOA emphasizes the importance of loose coupling. CSPA Services are independent, that is, they do not depend directly on each other. Organizations will need a technology solution to support communication between CSPA Services. This solution (for example a communication platform) will not affect the interfaces. It should be noted that SOA is not the same as Web Services, although they are often used in SOA.

102. In the discussion that follows, it is important to note that the Application Architecture is a descriptive (not prescriptive) framework that must support flexible approaches within individual statistical organizations while providing a basis for collaboration and sharing of standardized service components.

103. The CSPA Application Architecture must provide a means of expressing statistical production as an assembly of statistical services, aligned with design principles and guidelines and taking advantage of design patterns.

104. CSPA Services can be one of two types – a statistical function service or a statistical entity service.

- **Statistical function services** process statistical data based on statistical methodology. Statistical function services might for example include sampling services, imputation services and disclosure control services.
- **Statistical entity services** are an important and related set of services. These services provide reliable access to statistical information entities (objects) in order to support statistical production processes. Sometimes statistical entity management is included in a statistical function service. However, sometimes the entities are managed outside the service. Statistical entity services might include:
  - **Classification services** - for the management and use of statistical classifications
  - **Register services** - for the management and use of business, address, and household register information
  - **Geography services** - for the management and use of geographic information
  - **Statistical metadata services** - for the management and use of relevant statistical metadata throughout GSBPM statistical production

105. The IT departments of Statistical organizations will also be concerned with IT-specific services, which we term **Utility Services**, that provide important reusable service components of a non-statistical nature to create solutions. These are out of scope of CSPA, but sharing of common utility services where useful is encouraged amongst the statistical community.

106. The Figure 5 below shows the relationship between these "layers". The CSPA Application Architecture supports the identification of services, their definition, specification, and implementation, and their assembly into statistical production solutions. It is focused on expressing the statistical task (function) and entity services and how they are combined, under the direction of the business process layer which represents an instance of a GSBPM context (or alternatively, through an event-driven approach to be discussed below). Note that the bottom two layers are part of IT Community Sharing (libraries, tools, experiences) and are not part of the CSPA Application Architecture - they do in part touch on elements that may be part of the CSPA Technology Architecture.
107. Statistical services must be defined at an appropriate level of granularity. In Section III B (Business Architecture) we describe the following guidance in the section on design principles:

- CSPA Services are defined at an appropriate level of granularity […] defined at the level of a GSBPM sub-process they support, and
- CSPA Services are relevant to the business […] large enough for the business to understand, and are not low-level services used by IT.

108. An alternative approach to determining the size and complexity of services is microservices. Microservices are small, specialized and autonomous services, which collaborate to complete a process. Given their characteristics, microservices are well suited to enable a more fine-grained development of the application landscape to achieve business value.

109. One of the advantages of this approach is the granularity that can be obtained when scaling only services that have a high demand. From the example, if an organization has a huge demand to collect information from multiple sources it only needs to create new instances of the collect service, and not the others so demand of resources is more efficient. Using this approach will add further requirements to the production platform.

A. Contextual requirements for sharing and reuse
110. For services to be sharable and reusable, we make some assumptions as to the context in which services will be used. These assumptions follow from the business architecture and information architecture described in earlier chapters. The shareability of a service not only depends on the service itself, but also on the degree in which the contextual requirements are met.

111. The process context is described in the GSBPM. Statistical services do not need to serve just any kind of process, but they should serve GSBPM processes specifically.

112. The context in terms of data and metadata is described in a logical sense by GSIM. This means that services can “pick up” data and (configuring) metadata from the context in which they run.

113. The context is organised into statistical programs and statistical program cycles. This structure normally sets the context in which a service is being used. For example - the validation rules that are used for micro data editing often differs between different statistical programs but within a statistical program, the validation rules should be applied on all data regardless if the data comes from different exchange channels.

114. The context supports secure data management. Identity, and access management is central to the concept of being able to establish traceability and accountability. Roles needs to be attached to identities so services know what data can be accessed and what actions could be permitted. Data classification allows for knowing the proper security level that is needed to protect the data and to prevent data breaches and violations of the statistical confidentiality.

115. The context should allow for human adoption. When integrating a statistical service in an existing statistical organization, we care as much for IT integration as for the adoption by human organization. This includes workflow integration, user training and change management in general.

116. The context should offer proper governance. This means that there can be control over the process that brings changes to a statistical production environment. Depending on if it’s a hosted service or just code-sharing for a service the process for governance should be different. For a hosted service there must be a process that leads to a solution within an acceptable time period if problems arise. Hosted services should also have a defined process and strategy for its release management. Services shared as code should have a defined process for handling issues and feature request.

117. The organizational context should address “not invented here” or trust related issues. A specific mindset towards standard and open shareable services should exist. A culture should be built towards the production and integration of CSPA services, that addresses the trust issue that might exist when considering the adoption of a solution developed by another statistical organisation.

B. Features that enable sharing and reuse

118. Paragraph 27 introduced the concept of “wrapping” to enable existing tools to be transformed into CSPA services that can be more easily integrated. These wrappers, like services that are designed from scratch according to CSPA principles, will have specific features that support increased sharing and reuse.

119. Figure 6, below, provides an example of this, where a piece of core logic, which is the essence of the statistical service, is expanded with integration features. In this example we have expanded our core service with three features: documentation, multilingual support and published it as open source. Consider an algorithm, changing its input and output structures from something like numeric values and arrays to GSIM-structured input and output such as Codelists and Datasets is an example of adding a feature that eases integration into a statistical production process.

![Shared Service Diagram](image)

**Figure 6: Example - Expanding core logic with features**

**Metadata Driven**

120. The internal model used in the service can be in any type of model, and it should not require a complete rewrite of the internal core (as demonstrated in Figure 6 above). As long as it’s external facing interfaces exposes GSIM. This can be done using CSPA Adapters.
121. CSPA Services should have a common understanding of how the different information objects are structured, both for the input to the service and for the output of the service. Using the GSIM for guidance on how to represent data and metadata is key to creating services that can interoperate, and work in any statistical organization that uses GSIM. Parameters should also be properly described in a machine readable, and standardized way.

122. CSPA services should be context aware. This means that a service knows in which context it is being used and can pass along that information about its context to another service. For example, when a service is context aware it knows about what statistical program and statistical program cycle that it executes in. If needed it could call another service that is context aware and that service would seamlessly continue executing in the same context as the calling service in the appropriate statistical program and statistical program cycle, this is demonstrated in Figure 7 below.

123. CSPA Services should be layered so that they isolate the core logic of the service (the part of the service that delivers the business value) from modules meant for integrating the core logic into a specific environment. This makes the service easier to test, and easier to change, and it greatly eases portability of a service between statistical organizations. Examples of adapters can be adapters that binds to a specific internal data source and translate it into the services internal model. CSPA Adaptors are also a key to facilitate interoperability.

124. Support for Centralized data; some organizations might work with centralized data, meaning that all data (in resting states or even more general) is accessible in a uniform way. This can be implemented by having a data warehouse (which physically stores all data in the same location) or by employing data virtualization techniques, leaving the data where it is and allowing for easy connections to it. Features that support Centralized Data may easily hook up to centralized data solutions.

125. Support for Decentralized data; in case of decentralized data, data is confined to statistical domains or stovepipes. Data is not available in a uniform way. This means that there may be more need for adapters that perform data transformations to support integration once data is needed by services. Features that support Decentralized Data may easily hook up to centralized data solutions.

126. CSPA services should be able to work within different integration patterns. The following patterns are identified as commonly used:

- Point-to-point
- Orchestration
- Event-driven, Publish/Subscribe

127. All integration patterns require either synchronous or asynchronous interaction between the services, message brokers, orchestration engines, etc. By implementing relevant adapters that supports these interactions, a CSPA service will fit into any chosen pattern, or mix of patterns. A service core may by design be built to generate events from state changes.

128. Within statistical production, multiple patterns will be used to solve different problems, for example, an event-driven approach may be suitable for metadata service integrations while point-to-point may be suitable for integrations where large datasets are exchanged.

129. The CSPA Services must be able to support input and output in multiple languages where applicable. If the service includes manual operations, it also must be possible to change the language of the GUI. Preferably, even the presentation style should be adaptable. Code and comments should be in English.

130. CSPA Service Security needs to address the following key concerns:

- A service implementation must not contain any internal vulnerabilities that might pose risks to a local Statistical organisation environment
The statistical information (data, metadata) must be able to be protected at a level consistent with the needs of the local Statistical organisation. The service must not be vulnerable to attack where it may be used for wrong purposes.

131. In order to address information access, it is necessary to validate the user (invoker) through authentication, and to address access privileges through authorization controls.

132. The challenge in implementing security features in a service lies in the fact that in general different Statistical Organisations will have different (but similar) approaches.

133. Invoker Authentication may involve validating the identity of a user (person) or an invoking service. These can be mediated with certificates (tokens) or directory services (for users).

134. Access authorization involves resolving the rights of the invoker (person or service) and granting access to the relevant information and service resources based on these rights. The rights may be managed centrally (e.g. in Active Directory) or local to the service (in internal rights resources).

135. For the purpose of this document, the security concern relates to controls that are put in place to mitigate the risk that a CSPA Service or the data it controls is misused. This section provides some basic guidance on some of these controls. However, in general it is strongly advised that each CSPA Service implementation complete a Risk Assessment and document a Risk Mitigation Plan for high and extreme risks identified in the assessment.

Data at rest

136. Data at rest is of particular interest when a Statistical Service needs to defer state (see discussion in "Service Statelessness" in section V D). Under this circumstance, the security (e.g. encryption requirements or access control) of the data are entirely the responsibility of the Statistical Service. Where a CSPA Service already has a functional dependency on underlying technologies or platforms, it would be reasonable to make use of security functions available in those technologies.

Data in transit

137. Security of data in transit (e.g. contained within a message flow as part of a service invocation) will be considered in future iterations of CSPA.

Data Sensitivity

138. Sensitivity of statistical data varies amongst organizations and at this stage the architecture does not attempt to converge on a standard definition or treatment.

Machine to machine certification

139. Guidance for this will come in a future iteration of CSPA. Organization specific implementations based on assembly time infrastructure can assure security for service communication (for example, use of a VLAN).

Design principles

140. Enable services to be loosely coupled externally and be aware of internal coupling.

141. Maximize service autonomy (completeness) to enable share-ability and reusability (external & internal).

Open Source

142. CSPA Services should be distributed as open source with an appropriate license approved by the Open Source Initiative, depending on organizational policy. This provides transparency and visibility and promote collaboration, easier distribution, and the ability to create automated deploy pipelines.

Versioning

143. Different versions of services should be numbered based on Semantic Versioning.

Deployment

144. Virtualization will help shareability in multiple ways. It will address problems with shareable services that relies on different platforms and operating systems. Virtualization is also basis of other features like Sandboxing and Containerization.

145. For a CSPA service to be executable in any computing environment the concept of containers could be used. Dependencies on underlying technology and infrastructure are eliminated which increases the number of hosting options for service consumers. This allows for cloud hosting as well as flexible on-premise infrastructure alternatives.

146. CSPA Services should be able to be distributed and run in multiple instances, for scaling and security reasons. It will also help with rolling out updates without downtime.

Sandboxing for exploration
147. CSPA Services should be able to be sandboxed. This can be done in several ways. One way is that the service is packaged in a way that allows statistical organisations to explore the service in their environment. This means that the package could generate databases and runtime components automatically. Other ways to sandbox the services are to make the service runnable in an open environment so that statistical organisations don’t even need to download the service to explore its functionality. Setting up test environment for the service should also be simple for the consuming statistical organisations.

Performance

148. No specific guidance is provided on the performance characteristics. However, they should be declared in the CSPA Service Implementation Description and it is recommended that examples of performance level are included.

149. Some services could require interaction with large or unstructured data sources. This would require flexible hosting options to ensure that processing could take place as close to the data source as possible. Also, this might have impact on the programming of the service.

150. A CSPA Service will generally capture metrics related to the function that it performs. To all intents and purposes, these process metrics, which are used for reporting and auditing, are treated by the CSPA Service as just one of its outputs and should be reflected as such in the CSPA Service Specification.

Error Handling

151. Error handling, in this case, relates to situations where the service fails. The service must report this to the communication infrastructure if applicable. Error handling is left to the communication platform to handle as required. Generally, there will be protocol specific requirements for flagging errors. The error codes and their meanings need to be documented in the CSPA Service Implementation Description.

C. Documentation

152. The level of reusability promised by the adoption of a SOA is dependent on standardized documentation of the services. Services must be documented at least in English. CSPA has three layers to the description of any service.

153. In general, there will be one Service Specification corresponding to a Service Definition, to ensure that standard data exchange can occur. However, it is recognised that there may be occasions where an additional Service Specification is required, it is likely that this will be associated with variations in the methodology encapsulated within the statistical service. At the implementation level, services may have different implementations (software dependencies, protocols, supported methodologies) reflecting the environment of the supplying organization. Each implementation should adhere to the data format specified in the Service Specification.

These layers are described in the following paragraphs and in Figure 8. Completed examples are available within the CSPA catalogue, a specific example is the VTL service from Statistics Norway.

CSPA Service Definition

154. The CSPA Service Definition is at a conceptual level. In this document, the capabilities of a Statistical Service are described in terms of the GSBPM sub process that it relates to, the business function that it performs and GSIM information objects which are the inputs and outputs. A template of a CSPA Service Definition can be found in Annex 1.

CSPA Service Specification

155. The CSPA Service Specification is at a logical level. In this layer, the capabilities of a CSPA Service are fleshed out into business functions that have GSIM implementation level objects as inputs and outputs. This document also includes metrics and methodologies. A template of a CSPA Service Specification can be found in Annex 1.

CSPA Service Implementation Description

156. The CSPA Service Implementation Description is at an implementation (or physical) level. In this layer, the functions of the CSPA Service are refined into detailed operations whose inputs and outputs are GSIM implementation level objects.

157. This layer fully defines the service contract, including communications protocols, by means of the Service Implementation Description. It includes a precise description of all dependencies to the underlying infrastructure, non-functional characteristics and any relevant information about the configuration of the application being wrapped, when applicable. A template of a CSPA Service Implementation Description can be found in Annex 1.

158. Figure 8 shows the interfaces at different levels of abstraction.
There are a number of roles identified in CSPA (see Section VII) which are involved in the definition, specification, and implementation of CSPA Services. These roles can be considered as operating at different levels of abstraction. Figure 8 illustrates the relationship between these levels and roles for instances where there is one Service Specification for one Service Definition. Figure 9 illustrates the case where more than one Service Specification is required for one Service Definition.

**NOTE:** Open source technology is preferred for CSPA services, however it is recognized that statistical organizations have and continue to use proprietary software.
1. Paragraph 27 in 2.0 Common Statistical Production Architecture

2. Once all the externally facing adapters are ported to match the adopting NSIs infrastructure, this model also provides a to do list for the engineering teams re-implementing a published CSPA service locally.

3. Semantic Versioning. Given a version number MAJOR.MINOR.PATCH, increment the:

- MAJOR version when you make incompatible API changes,
- MINOR version when you add functionality in a backwards compatible manner, and
- PATCH version when you make backwards compatible bug fixes.

Additional labels for pre-release and build metadata are available as extensions to the MAJOR.MINOR.PATCH format.

https://semver.org/

4. https://www.statistical-services.org

5. https://www.statistical-services.org/o/rest/catalog/ng/service-detail?id=37;dv=36;sv=35;iv=24