Understanding TOGAF

About Functions and Capabilities

More about the structured approach to business architecture that underpins TOGAF and its artefacts

You may show this slide show without the written permission of the copyright holder, provided you also show you are opening it from the relevant avancier.website page.
The TOGAF content framework is not currently consistent

Principles: If an association appears in an artifact, it is a candidate for the meta model.

If an association does not appear in an artifact, it is a candidate for removal.

The important Function <is bounded by> Service association does not feature in the SOA chapter.
The important Process <realises> Service association is not represented in an Artefact.
The Org/Function matrix (important to structured analysis) is mentioned in the text, and reflected in the meta model, but does not appear in an Artefact.
The meta model lacks the Data Entity-Function association represented in an Artefact.
The meta model lacks the Service-Product association represented in an Artefact.
Actors and Organisation Units are distinct entities in the meta model, yet Actors are defined as including Organisation Units.
Actors being individuals, several associations to Org Unit or Actor would be better to Function or Role.
How TOGAF’s artifacts document a Business Architecture

- Principles catalogue
- Business Service/Product catalogue
- Process/Event/Control/Product catalogue
- Process flow diagram
- Data Entity catalogue
- Data Entity/Business Function matrix
- Driver/goal/objective catalogue
- Business Function/Service catalogue
- Functional decomposition
- Org. Function matrix
- Business Interaction matrix
- Organisation decomposition
- Organization/Actor catalogue
- Role catalogue
- Actor/Role matrix
TOGAF suggests 3 approaches to Business Architecture

- **Process Modelling**
  - Define high-level Processes/Scenarios
  - Decompose Processes to activities
  - Map activities to Roles & Functions

- **Structured Analysis**
  - "Identifies the key business Functions within the scope of the architecture, and maps those Functions onto the Org units within the business."

- **Use-case Analysis**
  - Identifies where Roles use Applications in performing Process steps, and defines the required IS (App) Services

“The level and rigor of decomposition needed varies from enterprise to enterprise”
Principles that underpin Structured Analysis, TOGAF and its artefacts, and relate Functions to Capabilities

1. Functions are independent of Organisation structure (8.4.1, 8.5).
2. Functions impose a structure on Activities sequenceable in Processes (34.2.1).
3. Functions/Capabilities are defined by Services provided (35.6.3)
4. Functions are used to describe Capabilities (34.2.1).
1) Functions are independent of Organisation structure

Structured Analysis
- “Identifies the key business Functions within the scope of the architecture, and maps those Functions onto the Org units within the business.” (8.4.1)

Business architecture outputs include
- “Correlation of Organization and Functions — relate business Functions to Organizational units in the form of a matrix report.” (8.5)

Organization Unit:
- “A self-contained unit of resources with goals, objectives, and measures.” (34.2.1)

1 Function/Capability may be mapped to 1 Org Unit, several Org Units, or parts of several.

Those Org Units inherit some or all of any target qualities given to Functions/Capabilities.
2) Functions impose a structure on Activities sequenceable in Processes

Atomic activities in Business Processes

May be placed at the *bottom* of the Functional Decomposition

**ArchiSurence: Function Hierarchy / Capability Map**

- **Customer Relations**
  - Open Policy
  - Record Claim
  - Email Approval

- **Claim Handling**
  - Assess Claim

- **Finance**
  - Pay Claim
  - Bank Premium

- **Product Management**
  - Policy Definition
  - Legal Compliance
  - Maintain client records

A strict (non-redundant) hierarchy.
In theory, every atomic activity in a Process can be placed at the bottom of a Functional decomposition hierarchy (34.2.1).

In practice
- The Function hierarchy usually stops at a high (3rd or 4th) level
- Some Process models descend to a lower (5th or 6th) level.
3) Functions/Capabilities are defined by Services provided

“The purpose of the Business Service/Function catalog is to provide a Functional decomposition…[It] can be used to identify Capabilities of an Organization…. “(35.6.3)

Business architecture outputs (8.5) include

- “Business Functions — a detailed, recursive step involving successive decomposition of major Functional areas into sub-Functions.”
- “Correlation of Organization and Functions — relate business Functions to Organizational units in the form of a matrix report.”
Capabilities as Functions

- Function describes units of business Capability at all levels of granularity (TOGAF 34.2.1)
- Capabilities are typically expressed in general and high-level terms and typically require a combination of Organization, people, Processes, and technology to achieve. For example, marketing, customer contact, or outbound telemarketing. [cf. Function Names] (TOGAF 3.26)
- This Functional decomposition can be used to identify new Capabilities required to support business change.
- The purpose of the Functional Decomposition diagram is to show on a single page the Capabilities of an Organization…. (TOGAF 35.6.3)

- Implication: Functions have all the attributes Capabilities have - including target qualities
Chapter 32 on Capability-Based Planning has 10 (ten!) references to Capabilities being cross-organisational.

- It is not about improving an Organisation Unit
- It is about improving a named Function (say, HR) regardless of where it is carried out in the Organisation

Both Structured Analysis and CBP encourage architects to
- discuss Functions/Capabilities independently of Org Units.
- examine business objectives, services and processes before mapping those to Organisation Units.

“Function describes units of business Capability at all levels of granularity” TOGAF 34.2.1
EA is about human and computer activity systems in which Actors/components achieve desired effects by playing Roles in Processes.

A required **Capability** (say Sales, or Disaster Recovery) can be specified as a group of
- required products or **Services** resulting from **Processes** performed by **Actors/components**

A required **Function** (say Sales, or Disaster Recovery) can be specified as a group of
- required products or **Services** resulting from **Processes** performed by **Actors/components**
A Capability is a **view** rooted in a Function (at whatever level of granularity is chosen).

It can encompass as many of the remaining entities as you choose.
So, Capability = Function ++

- In business architecture documentation

- Capability (say Marketing) =
- Function (Marketing) + quality targets + resources needed
Reader’s challenge 1: What makes Disaster Handling a Capability?

► An enterprise may not formalise disaster handling
  ■ It may rely on the ad hoc responses of intelligent human Actors when a disaster happens.
  ■ It may do little or nothing other than encourage Actors to think about disasters and what they can do about them.

► Or, it may set out to develop a disaster handling Capability
► Which is to formalise and systemise
  ■ Nominate a DR Function
  ■ Define what kinds of disaster are to be anticipated
  ■ Define targets for recovery from those disasters
  ■ Define Roles and Processes needed to achieve DR
  ■ Acquire resources (Actors, components etc.) needed to perform DR Roles and Processes
  ■ Testing
  ■ Etc.
Reader’s challenge 2: What makes Innovation a Capability?

► You may perceive an enterprise as being innovative
► But so far, innovations have been down to human inspiration and motivation to follow them up.

► How to develop a tangible innovation “Capability”?
► The EA concept of Capability implies a degree of systemisation
  ■ Nominate an innovation Function
  ■ Define where innovation is sought
  ■ Define targets for innovation success (e.g. income from new products)
  ■ Define Roles (if not Processes) needed to achieve innovation
  ■ Acquire resources (Actors) needed perform the Roles and Processes
  ■ Etc.
Variations of "Capability" appear in “Capability maturity models” and “Capability based planning”, and in TOGAF (with various meanings)

DoDAF is built around Capability-based planning. It does not have the concept of a business Function, because Capability takes its place.

Some Capabilities may correspond to a Function in a primary business-as-usual Functional decomposition hierarchy (usually but not inevitably a high-level Function).

Other Capabilities (e.g. “Disaster Handling”) might not appear in the primary Function hierarchy. But you can define other Function hierarchies.

And you can define a Function independently of any hierarchical decomposition structure, as a free-standing structural component, a grouping of any activities you choose.

So, whether your Capability is named in a Function hierarchy or not, it is always representable as a Function + target qualities + resources needed.
Avancier Methods
Abstracting a meta from EA artefacts

Based on the structured approach to business architecture that underpins TOGAF and its artefacts

You may show this slide show without the written permission of the copyright holder, provided you also show you are opening it from the relevant avancier.website page.
How TOGAF documents a business activity system

If an association does not appear in an artifact, remove it (or else find an artefact).

E.g. which TOGAF products record these two associations?
Skills Framework for the Information Age suggests 7 views of EA

- The Views can be centred on the Atomic Business Activities
Designing activity systems

- Define required Services, and Processes to deliver them
- Assign activities in Processes to Functions and Roles
- Find Org units and Actors to perform the activities when required

<table>
<thead>
<tr>
<th>Passive structure acted on</th>
<th>Required behaviour elements triggered by events, produce results</th>
<th>Logical active structure elements group activity types</th>
<th>Physical active structure elements perform activities</th>
</tr>
</thead>
<tbody>
<tr>
<td>Business</td>
<td>Business Activity</td>
<td>Function</td>
<td>Org Unit</td>
</tr>
<tr>
<td></td>
<td>Process</td>
<td>Functional Decomposition</td>
<td>Organisation Structure</td>
</tr>
<tr>
<td></td>
<td>Process Flow diagram</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Data Entity/ Business Function matrix</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Information Systems</td>
<td>IS (App) Service</td>
<td>Application Interface</td>
<td>Application</td>
</tr>
<tr>
<td></td>
<td>IS Service catalogue</td>
<td></td>
<td>Application Portfolio Cat</td>
</tr>
<tr>
<td></td>
<td>Data Entity</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Copyright Avancier Limited 2007 - 2016
### Design sequences used in EA

**Function:** externally, a group of service types; internally, a group of activity types

**Role:** externally, a group of service types; internally, a group of activity types

<table>
<thead>
<tr>
<th>Business</th>
<th>Logical Structure</th>
<th>“Physical” Structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Business Service</td>
<td>Function</td>
<td>Org Unit</td>
</tr>
<tr>
<td>Business Process</td>
<td>Role</td>
<td>Actor</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Data</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>IS (App) Service</td>
<td>Data Model</td>
<td>Data Store</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Applications</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Platform Service</td>
<td>Application Interface</td>
<td>Application</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Technology</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Platform Service</td>
<td>Platform Interface</td>
<td>Platform Application</td>
</tr>
</tbody>
</table>

Nothing is really physical in EA, but when implemented, a physical element must be addressable.
### Core artifacts:
describe architectural entities and relations between them

<table>
<thead>
<tr>
<th>Section</th>
<th>Artifacts</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>2 Context / Precursors</strong></td>
<td>Driver/Goal/Objectives catalogue, Principles catalogue, Stakeholder catalogue, Requirements catalogue, Business Scenario diagram, Solution Vision diagram</td>
</tr>
<tr>
<td><strong>5 Data Architecture</strong></td>
<td>Data Entity catalogue, Data Entity/Business Function matrix, Data Store catalogue, Logical Data Model diagram, Data Dissemination diagram, CRUD diagram</td>
</tr>
<tr>
<td><strong>7 Applications Architecture</strong></td>
<td>Information Services catalogue, Application Portfolio catalogue, Data Flow catalogue, Application/Data Entity matrix, Application/Function matrix, Application Communication diagram, Application Use Case diagram, Application Platform Service catalogue</td>
</tr>
<tr>
<td><strong>10 Migration Planning</strong></td>
<td>RAID catalogue, Value/Cost/Risk Grid, Migration Path, Road Map</td>
</tr>
</tbody>
</table>
Abstracting a meta model from the artifacts

Driver/goal/objective catalogue
Principles catalogue
Requirements catalogue
Process catalogue
Process Map diagram
Process Flow diagram
Data Entity/Business Function matrix
Data Entity catalogue
Use Case diagram
Info Services catalogue
Application Platform Service catalogue
Technical Reference Model

Business

Business Service
Function
Org Unit
Process
Role
Actor

Data / Information

Data Entity
Data Store

Applications

Info. Service
Application Interface
Application

Infrastracture

Platform Service
Platform Interface
Platform Application

Copyright Avancier Limited 2007 - 2016
As complete as seems reasonable
Avancier Methods are useful with all architecture frameworks that share similar ends and means.

- BCS E&SA reference model
- TOGAF The Open Group
- ArchiMate Language Framework
- IBM’s view EA
- CSC’s domains of change (POLDAT)
- EA as Strategy” MIT