*Chapter 5*

# *Business Scenarios and Business Goals*

This handout is extracted from TOGAF 9.1 Chapter 26, supporting Business Scenarios, a method for deriving business requirements for architecture and the implied technical requirements. This handout can be used for TOGAF 9 Level 1 and Level 2 training.

## 5.1 Developing Business Scenarios

### 5.1.1 General Guidelines

The stakeholders (e.g., business managers, end users) will tell you what they want, but as an architect you must still gain an understanding of the business, so you must know the most important actors in the system. If the stakeholders do not know what they want:

- Take time, observe, and record how they are working today
- Structure information in such a way that it can be used later
- Uncover critical business rules from domain experts
- Stay focused on what needs to be accomplished, and how it is to be accomplished

This effort provides the anchor for a chain of reason from business requirements through to technical solutions. It will pay off later to be diligent and critical at the start.

### 5.1.2 Questions to Ask for Each Area

The business scenario workshops mentioned above in the Gathering phase are really structured interviews. While there is no single set of appropriate questions to ask in all situations, the following provides some guidance to help business scenario consultants in asking questions.

**Identifying, Documenting, and Ranking the Problem**

Is the problem described as a statement of *what* needs to be accomplished, like steps in a process, and not *how* (with technology "push")?

If the problem is too specific or a "how":

- Raise a red flag
- Ask "Why do you need to do it that way?" questions

If the problem is too vague or not actionable:

- Raise a red flag
- Ask "What is it you need to do, or will be able to do if this problem is solved?" questions

Ask questions that help to identify where and when the problem exists:

- Where are you experiencing this particular problem? In what business process?
- When do you encounter these issues? During the beginning of the process, the middle, the end?

Ask questions that help to identify the costs of the problem:

- Do you account for the costs associated with this problem? If so, what are they?
- Are there hidden costs? If so, what are they?
- Is the cost of this problem covered in the cost of something else? If so, what and how much?
- Is the problem manifested in terms of poor quality or a perception of an ineffective organization?

### Identifying the Business & Technical Environment, and Documenting in Models

Questions to ask about the business environment:

- What key process suffers from the issues? What are the major steps that need to be processed?
- Location/scale of internal business departments?
- Location/scale of external business partners?
- Any specific business rules and regulations related to the situation?

Questions to ask about the current technology environment:

- What technology components are already presupposed to be related to this problem?
- Are there any technology constraints?
- Are there any technology principles that apply?

### Identifying and Documenting Objectives

Is the "what" sufficiently backed up with the rationale for "why"? If not, ask for measurable rationale in the following areas:

- Return on investment
- Scalability
- Performance needs
- Compliance to standards
- Ease-of-use measures

### Identifying Human Actors and their Place in the Business Model

An actor represents anything that interacts with or within the system.  This can be a human, or a machine, or a computer program. Actors initiate activity with the system, for example:

- Computer user with the computer
- Phone user with the telephone
- Payroll clerk with the payroll system
- Internet subscriber with the web browser

An actor represents a role that a user plays; i.e., a user is someone playing a role while using the system (e.g., John (user) is a dispatcher (actor)). Each actor uses the system in different ways (otherwise they should be the same actor). Ask about the humans that will be involved, from different viewpoints, such as:

- Developer
- Maintainer
- Operator
- Administrator
- User

### Identifying Computer Actors and their Place in the Technology Model

Ask about the computer components likely to be involved, again from different points of view. What must they do?

### Documenting Roles, Responsibilities, Measures of Success, Required Scripts

When defining roles, ask questions like:

- What are the main tasks of the actor?
- Will the actor have to read/write/change any information?
- Will the actor have to inform the system about outside changes?
- Does the actor wish to be informed about unexpected changes?

### Checking for Fitness-for-Purpose, and refining if necessary

Is there enough information to identify who/what could fulfil the requirement? If not, probe more deeply.

Is there a description of when, and how often, the requirement needs to be addressed? If not, ask about timing.

## 5.2 Business Scenario Documentation

### 5.2.1 Textual Documentation

Effective business scenario documentation requires a balance between ensuring that the detail is accessible, and preventing it from overshadowing the results and overwhelming the reader. To this end, the business scenario document should have the main findings in the body of the document and the details in appendices.

In the appendices:

- Capture all the important details about a business scenario:
  - Situation description and rationale
  - All measurements
  - All actor roles and sub-measurements
  - All services required
- Capture the critical steps between actors that address the situation, and sequence the interactions
- Declare relevant information about all actors:
  - Partition the responsibility of the actors
  - List pre-conditions that have to be met prior to proper system functionality
  - Provide technical requirements for the service to be of acceptable quality

In the main body of the business scenario:

- Generalize all the relevant data from the detail in the appendices

### 5.2.2 Business Scenario Models

- Remember the purpose of using models:
  - Help comprehension
  - Give a starting point to confirm requirements
  - Relate actors and interactions
- Keep drawings clear and neat:
  - Do not put too much into one diagram
  - Simpler diagrams are easier to understand
- Number diagrams for easy reference:
  - Maintain a catalog of the numbers to avoid duplicates

## 5.3 Guidelines on Goals and Objectives

### 5.3.1 Importance of Goals

One of the first steps in the development of an architecture is to define the overall goals and objectives for the development. The objectives should be derived from the business goals of the organization, and the way in which IT is seen to contribute to meeting those goals.

Every organization behaves differently in this respect, some seeing IT as the driving force for the enterprise and others seeing IT in a supporting role, simply automating the business processes which already exist. The essential thing is that the architectural objectives should be very closely aligned with the business goals and objectives of the organization.

### 5.3.2 Importance of SMART Objectives

Not only must goals be stated in general terms, but also specific measures need to be attached to them to make them SMART, as described above.

The amount of effort spent in doing this will lead to greater clarity for the sponsors of the architecture evolution cycle. It will pay back by driving proposed solutions much more closely toward the goals at each step of the cycle. It is extremely helpful for the different stakeholders inside the organization, as well as for suppliers and consultants, to have a clear yardstick for measuring fitness-for-purpose. If done well, the ADM can be used to trace specific decisions back to criteria, and thus yield their justification.

The goals below have been adapted from those given in previous versions of TOGAF. These are categories of goals, each with a list of possible objectives. Each of these objectives should be made SMART with specific measures and metrics for the task. However, since the actual work to be done will be specific to the architecture project concerned, it is not possible to provide a list of generic SMART objectives that will relate to any project.

Instead, we provide here some example SMART objectives.

#### Example of Making Objectives SMART

Under the general goal heading “Improve User Productivity” below, there is an objective to provide a “Consistent User Interface” and it is described as follows:

> “A consistent user interface will ensure that all user-accessible functions and services will appear and behave in a similar, predictable fashion regardless of application or site. This will lead to better efficiency and fewer user errors, which in turn may result in lower recovery costs.”

To make this objective SMART, we ask whether the objective is specific, measurable, actionable, realistic, and time-bound, and then augment the objective appropriately.

The following captures an analysis of these criteria for the stated objective:

- **Specific**: The objective of providing “a consistent user interface that will ensure all user accessible functions and services will appear and behave in a similar, predictable fashion regardless of application or site”. is pretty specific. However, the measures listed in the second sentence could be more specific . . .

- **Measurable**: As stated above, the objective is measurable, but could be more specific. The second sentence could be amended to read (for example): "This will lead to 10% greater user efficiency and 20% fewer order entry user errors, which in turn may result in 5% lower order entry costs".
- **Actionable**: The objective does appear to be actionable. It seems clear that consistency of the user interface must be provided, and that could be handled by whoever is responsible for providing the user interface to the user device.
- **Realistic**: The objective of providing "a consistent user interface that will ensure all user accessible functions and services will appear and behave in a similar, predictable fashion regardless of application or site" might not be realistic. Considering the use today of PDAs at the user end might lead us to augment this objective to ensure that the downstream developers don't unduly create designs that hinder the use of new technologies. The objective could be re-stated as "a consistent user interface, across user interface devices that provide similar functionality, that will ensure . . ." etc.
- **Time-bound**: The objective as stated is not time-bound. To be time-bound the objective could be re-stated as "By the end of Q3, provide a consistent . . .".

The above results in a SMART objective that looks more like this (again remember this is an example):

> "By the end of Q3, provide a consistent user interface across user interface devices that provide similar functionality to ensure all user accessible functions and services appear and behave in a similar way when using those devices in a predictable fashion regardless of application or site. This will lead to 10% greater user efficiency and 20% fewer order entry user errors, which in turn may result in 5% lower order entry costs."

### 5.3.3 Categories of Goals and Objectives

Although every organization will have its own set of goals, some examples may help in the development of an organization-specific list. The goals given below are categories of goals, each with a list of possible objectives, which have been adapted from the goals given in previous versions of TOGAF.

Each of the objectives given below should be made SMART with specific measures and metrics for the task involved, as illustrated in the example above. However, the actual work to be done will be specific to the architecture project concerned, and it is not possible to provide a list of generic SMART objectives that will relate to any project.

#### Goal: Improve Business Process Performance

Business process improvements can be realized through the following objectives:

- Increased process throughput
- Consistent output quality
- Predictable process costs
- Increased re-use of existing processes
- Reduced time of sending business information from one process to another process

**Goal: Decrease Costs**

Cost improvements can be realized through the following objectives:

- Lower levels of redundancy and duplication in assets throughout the enterprise
- Decreased reliance on external IT service providers for integration and customization
- Lower costs of maintenance

**Goal: Improve Business Operations**

Business operations improvements can be realized through the following objectives:

- Increased budget available to new business features
- Decreased costs of running the business
- Decreased time-to-market for products or services
- Increased quality of services to customers
- Improved quality of business information

**Goal: Improve Management Efficacy**

Management efficacy improvements can be realized through the following objectives:

- Increased flexibility of business
- Shorter time to make decisions
- Higher quality decisions

**Goal: Reduce Risk**

Risk improvements can be realized through the following objectives:

- Ease of implementing new processes
- Decreased errors introduced into business processes through complex and faulty systems
- Decreased real-world safety hazards (including hazards that cause loss of life)

**Goal: Improve Effectiveness of IT Organization**

IT organization effectiveness can be realized through the following objectives:

- Increased rollout of new projects
- Decreased time to rollout new projects
- Lower cost in rolling out new projects
- Decreased loss of service continuity when rolling out new projects
- Common development: applications that are common to multiple business areas will be developed or acquired once and re-used rather than separately developed by each business area.
- Open systems environment: a standards-based common operating environment, which accommodates the injection of new standards, technologies, and applications on an organization-wide basis, will be established. This standards-based environment will provide the basis for development of common applications and facilitate software re-use.

- Use of products: as far as possible, hardware-independent, off-the-shelf items should be used to satisfy requirements in order to reduce dependence on custom developments and to reduce development and maintenance costs.
- Software re-use: for those applications that must be custom developed, development of portable applications will reduce the amount of software developed and add to the inventory of software suitable for re-use by other systems.
- Resource sharing: data processing resources (hardware, software, and data) will be shared by all users requiring the services of those resources. Resource sharing will be accomplished in the context of security and operational considerations.

**Goal: Improve User Productivity**

User productivity improvements can be realized through the following objectives:

- Consistent user interface: a consistent user interface will ensure that all user-accessible functions and services will appear and behave in a similar, predictable fashion regardless of application or site. This will lead to better efficiency and fewer user errors, which in turn may result in lower recovery costs.
- Integrated applications: applications available to the user will behave in a logically consistent manner across user environments, which will lead to the same benefits as a consistent user interface.
- Data sharing: databases will be shared across the organization in the context of security and operational considerations, leading to increased ease-of-access to required data.

**Goal: Improve Portability and Scalability**

The portability and scalability of applications will be through the following objectives:

- Portability: applications that adhere to open systems standards will be portable, leading to increased ease-of-movement across heterogeneous computing platforms. Portable applications can allow sites to upgrade their platforms as technological improvements occur, with minimal impact on operations.
- Scalability: applications that conform to the model will be configurable, allowing operation on the full spectrum of platforms required.

**Goal: Improve Interoperability**

Interoperability improvements across applications and business areas can be realized through the following objectives:

- Common infrastructure: the architecture should promote a communications and computing infrastructure based on open systems and systems transparency including, but not limited to, operating systems, database management, data interchange, network services, network management, and user interfaces.
- Standardization: by implementing standards-based platforms, applications will be provided with and will be able to use a common set of services that improve the opportunities for interoperability.

### Goal: Increase Vendor Independence

Vendor independence will be increased through the following objectives:

- Interchangeable components: only hardware and software that have standards-based interfaces will be selected, so that upgrades or the insertion of new products will result in minimal disruption to the user's environment.
- Non-proprietary specifications: capabilities will be defined in terms of non-proprietary specifications that support full and open competition and are available to any vendor for use in developing commercial products.

### Goal: Reduce Lifecycle Costs

Lifecycle costs can be reduced through most of the objectives discussed above. In addition, the following objectives directly address reduction of lifecycle costs:

- Reduced duplication: replacement of isolated systems and islands of automation with interconnected open systems will lead to reductions in overlapping functionality, data duplication, and unneeded redundancy because open systems can share data and other resources.
- Reduced software maintenance costs: reductions in the quantity and variety of software used in the organization will lead to reductions in the amount and cost of software maintenance. Use of standard off-the-shelf software will lead to further reductions in costs since vendors of such software distribute their product maintenance costs across a much larger user base.
- Incremental replacement: common interfaces to shared infrastructure components allow for phased replacement or upgrade with minimal operational disturbance.
- Reduced training costs: common systems and consistent Human Computer Interfaces (HCIs) will lead to reduced training costs.

### Goal: Improve Security

Security can be improved in the organization's information through the following objectives:

- Consistent security interfaces for applications: consistent security interfaces and procedures will lead to fewer errors when developing applications and increased application portability. Not all applications will need the same suite of security features, but any features used will be consistent across applications.
- Consistent security interfaces for users: a common user interface to security features will lead to reduced learning time when moving from system to system.
- Security independence: application deployment can use the security policy and mechanisms appropriate to the particular environment if there is good layering in the architecture.
- A 25% reduction in calls to the help desk relating to security issues.
- A 20% reduction in "false positives" detected in the network (a false positive is an event that appears to be an actionable security event, but in fact is a false alarm).

**Goal: Improve Manageability**

Management improvement can be realized through the following objectives:

- Consistent management interface: consistent management practices and procedures will facilitate management across all applications and their underlying support structures. A consistent interface can simplify the management burden, leading to increased user efficiency.
- Reduced operation, administration, and maintenance costs: operation, administration, and maintenance costs may be reduced through the availability of improved management products and increased standardization of the objects being managed.