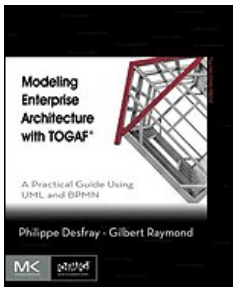

Modeling Enterprise Architecture with TOGAF: A Practical Guide Using UML and BPMN

by Philippe Desfray and Gilbert Raymond

Chapter 12: SOA, Processes, and Information

OVERVIEW

Beyond the context of TOGAF, this chapter looks at three major themes of enterprise architecture and information systems: the SOA approach, business processes, and information.

The first theme concerns information system agility and reuse. Even though Service-Oriented Architecture is no longer perceived today as being the universal solution, it does provide a set of good practices which should be taken into account.

Mastering business processes is a key element in enterprise architecture, at the very heart of changes made within organizations. Identification, modeling, and governance also widely influence architecture work.

Information constitutes the raw material of operations carried out by systems. From databases to messages, from documents to e-mails, the variety of forms of information and information management modes require adapted practices and organization modes in enterprise architecture.

12.1 SERVICE-ORIENTED ARCHITECTURE

12.1.1 SOA in TOGAF

SOA (Service-Oriented Architecture) is a style of architecture organized using common business services that are pooled for a set of business lines or applications.[a] Here we do not discuss so-called technical SOA, which consists of implementing a set of devices and technologies that focus on web services (ESB, UDDI, etc.), since even though these techniques really do facilitate communication between the components of the IS, the structure of the system remains the key architectural question, regardless of the technical means used.

The primary motivation of service-oriented architecture comes from the following observation: The breakdown into independent application silos (monolithic blocks) is one of the major sources of difficulty in system evolution and maintenance work.

Let's not forget that TOGAF does not advocate any particular architectural style. TOGAF presents SOA in the Chapter 22 "Using TOGAF to Define & Govern SOAs" in the ADM Guidelines and techniques part. Furthermore, The Open Group set up a working group dedicated to this subject, who published a specific document[b] alongside the TOGAF publications. Figure 12.1 summarizes the SOA vision developed in this document.

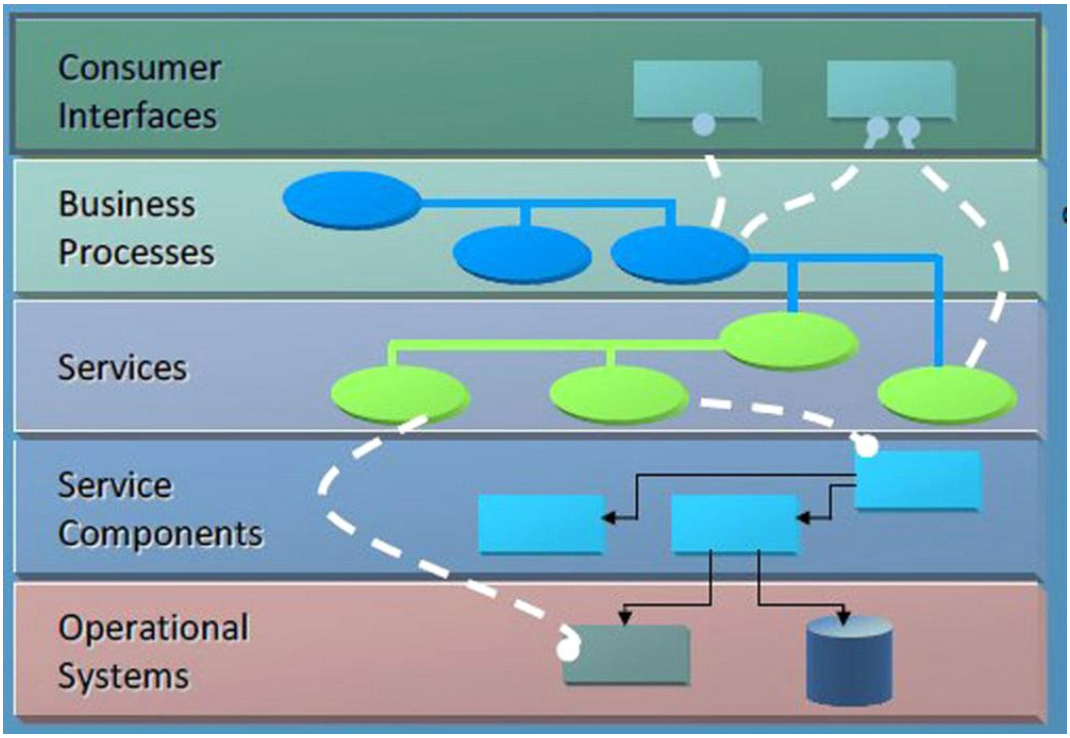

Figure 12.1: SOA structure according to "SOA Reference Architecture."

The system is broken down into five layers:

- The client interface

- Business processes
- Services
- Service components
- The system platform

We can see that the system is no longer broken down into "applications," but rather into components, each positioned in a particular layer that corresponds to a specific role in the system. However, this does not mean that applications disappear: they are built by combining a set of components, which are potentially pooled.

12.1.2 SOA: Not as Simple as it Seems

This organization aims to encourage system agility and to enable a high degree of service reuse. However, beyond a naive vision, the management of this type of architecture requires a solid understanding of its foundations and its difficulties.

There is a big difference between SOA architecture and the usual structure built from applications (which we can call "application-oriented architecture"). For the latter, the application is the fundamental constituent of the system, and this structure coincides with the organization of teams. We find here a breakdown into application silos, both at the heart of the system and in its management.

With an SOA-type breakdown, things are different. First, the basic constituent of the system (the system component) is much less high level than the application, and this mechanically increases the complexity of the system. For example, we go from managing 100 applications to managing 1000 service components. Second, this breakdown does not cover the historical organization by application, which remains a strong operational unit linked to business demands.

These questions require special, well-managed handling. The proliferation of disorganized services is one of the main pitfalls encountered, often due to overly technically oriented vision. Some people have no hesitation in talking about *spaghetti-oriented architecture*.[c]

12.1.3 Organizing Components

One of the best way to control this complexity is to structure components by differents types and levels. We have already seen this kind of approach advocated by TOGAF, which defines five levels (client interface, business processes, services, service components, system platform). This type of structuring is accompanied by rules and best practices. The main rule establishes a norm for dependencies, which forbids, for example, the use of a component by a higher layer (a service component cannot depend on a client interface).

Other typologies exist and are relatively similar, despite variations or different levels of detail. Table 12.1 presents some of these possible typologies:

Table 12.1: Different SOA Typologies

Herzum and Sims[a]	ESOA[b]	Microsoft[c]	IBM[d]
	Front-end application	Presentation layer	Presentation
Process	Process centric	Business process	Business process choreography
	Intermediary	Business service	Composite service
Entity	Basic	Data service	Service
Utility			

[a] Business Component Factory, Peter Herzum and Oliver Sims, Wiley Computing Publishing, 2000.

[b] Enterprise SOA, Dirk Krafzig, Karl Banke, and Dirk Slama, The Coad Series, 2005.

[c] An Overview of Service-Oriented Architecture in Retail, Moin Moinuddin, Microsoft, January 2007.

[d] Bernhard Borges, Kerrie Holley, and Ali Arsanjani, IBM, September 15, 2004, SearchWebServices.com.

In Chapter 9, we used a typology based on the same principles, which uses the suggestions made in the "Logical Architecture: Principles, structures and best practices" white paper (Architecture Logique: Principes, structures et bonnes pratiques[d]):

- *Interaction component*: Exchange with the outside world
- *Process component*: Process automation
- *Function component or intermediate component*: Business processing and data composition
- *Entity component*: Data access
- *Utility component*: Cross-organizational functionalities (messaging system, address book, etc.)

For the enterprise architect, the exercise consists of identifying and positioning each component within this framework, allocating well-defined services to each component, and specifying the execution conditions for these services in the form of a service-level agreement (SLA).

12.1.4 Encouraging Reuse

This objective is not new in the field of software. As we have just seen, it is one of the points highlighted in service-oriented architecture. However, experience shows that it is not always easy to concretely implement this goal. Sharing leads to dependency between different user parts, which creates additional management costs.

The term "reuse" can designate several realities.

Reuse Through Copying and Pooling

Reuse through copying consists of taking what already exists and using it in another context. In the context of software development, this comes down to duplicating a part of the source code from a given example. More generally, it is the widely known mechanism used in design "patterns." A tried and tested practice or structure is reused by adapting it to a specific context. In this case, the reused part is integrated and merged into the target component, which evolves with no direct link to the element used. As discussed in Section 4.1, the TOGAF architecture repository is the ideal place to store this kind of heritage, which must be constantly added to.

Shared reuse or *pooling* is very different. It consists of sharing a component in different contexts; this is the typical schema used in an SOA context, where several applications will use the services provided by a component deployed in the system. To use SOA terminology, users are *consumers*, and the shared component is the service *provider*. This schema develops links between consumers, which must be managed.

The Price of Pooling

On this issue, TOGAF considers that pooling components multiply costs by at least a factor of two compared with separate development.[e] The main causes of these additional costs are as follows:

- First, pooling requires agreement between consumers regarding the terms of the service provided. This contractualization (SLA) precisely defines the conditions under which the service in question will operate. Moreover, we can see that the nonfunctional part plays its full role here: availability, performance, security, and so on.
- Second, each provider evolution is immediately transmitted to all the user clients of the pooled component. This situation often leads to internal conflicts.

Let's take the example of two applications called A and B, which use the services of a single component named C. For specific reasons, application B needs to quickly modify the interface of component C. The person in charge of application A is forced to resume a test campaign if this component is modified, and to modify his work plan for external reasons.

Thus, we end up with the following paradox: the higher the degree of pooling, the more difficult contractualization becomes. Therefore, if reuse is not managed properly, it can lead to reduced system agility, bogged down by the red tape of contractualization.

Faced with this type of situation, IS managers sometimes decide to temporarily multiply the number of versions of pooled components in order to enable rapid deployment without perturbing other consumers. In Figure 12.2, two versions of component C will be available: version C1 (unchanged) used by application A, and version C2 integrating the modifications requested by application B. This transitional state has to be resolved by the effective pooling of the same version of C by A and B.

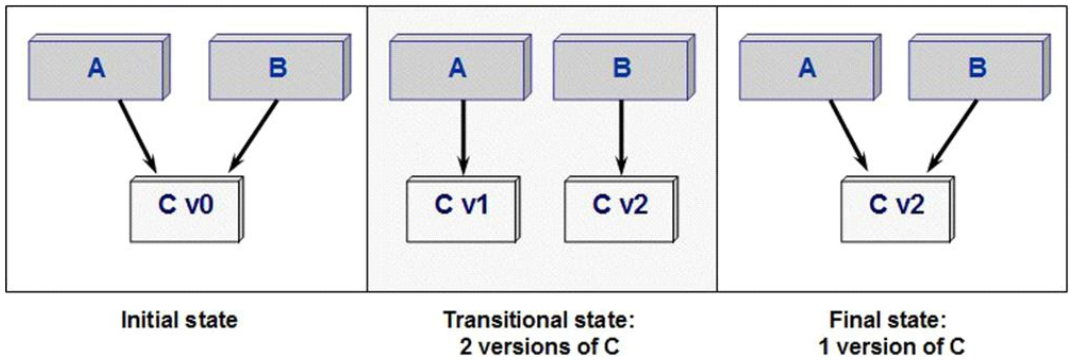

Figure 12.2: SOA component versioning

This option enables conflicts to be reduced, but leads to a multiplication of versions, which eventually compromises effective reuse and increases the complexity of the system.

For the sake of pragmatism and efficiency, these situations are difficult to forbid entirely. The role of the organization in charge of enterprise architecture consists of avoiding a proliferation of versions. An adapted indicator will facilitate control and management, for example, by fixing a maximum number of simultaneous versions and a limited duration for the coexistence of several versions of a component.

Note Beware of focusing exclusively on reuse. Many "reusable" components exist, but a lot fewer components are actually "reused." This is

> one of the most common aspects of "overengineering," which tends to minimize difficulties. We strongly recommend systematically designing reuse in concrete terms (in other words, with at least two uses) and avoiding anticipating potential reuse with no real implementation.

Thus, the work of an enterprise architect consists of finding the best compromise between the advantages of reuse and the disadvantages that we have just discussed.

This choice is based on a set of characteristics:

- The number of consumers
- The stability of the pooled component
- The frequency of use
- The size/complexity of the interface

In other words, for a component used by a large number of consumers, which is very unstable, frequently used, and relatively complex to approach, significant additional costs can be expected.

Stability plays a major role in this type of choice. It is often linked to a certain normalization of the service provided by the provider, which reduces considerably the burden of management. In this category we find many utility components, for example, pooling an e-mail server presents no particular difficulties, despite the fact that it is used by a large number of consumers to automatically send e-mail.

12.1.5 The BPM-SOA Couple

Process automation is becoming more and more widespread and constitutes a discipline in itself: BPM (*Business Process Management*),[f] with its community, methods, and tools. This tendency is also found in the main ERPs, which today include the process automation component in their solutions.

Two main categories of automated process can be identified: first, processes with a high level of human intervention, in other words, processes in which human actors are in charge of running tasks (using software applications, for example), and second, processes where there is little or no human intervention.

The first category is typically the field of workflows (historically workflow tools). The workflow engine takes charge of sequencing and allocating tasks to different participants, which means that it has to have knowledge of the organization and each person's capabilities. It acts as a "robot" manager which coordinates and manages the progress of the work.

The second category is likely to be simpler to implement, inasmuch as it deals with the sequencing of automatic operations, carried out by defined applications or software components. In this case, the term process "orchestration" is often used.

Figure 12.3 presents a typical architecture change, from an application organization to a BPM/SOA approach.

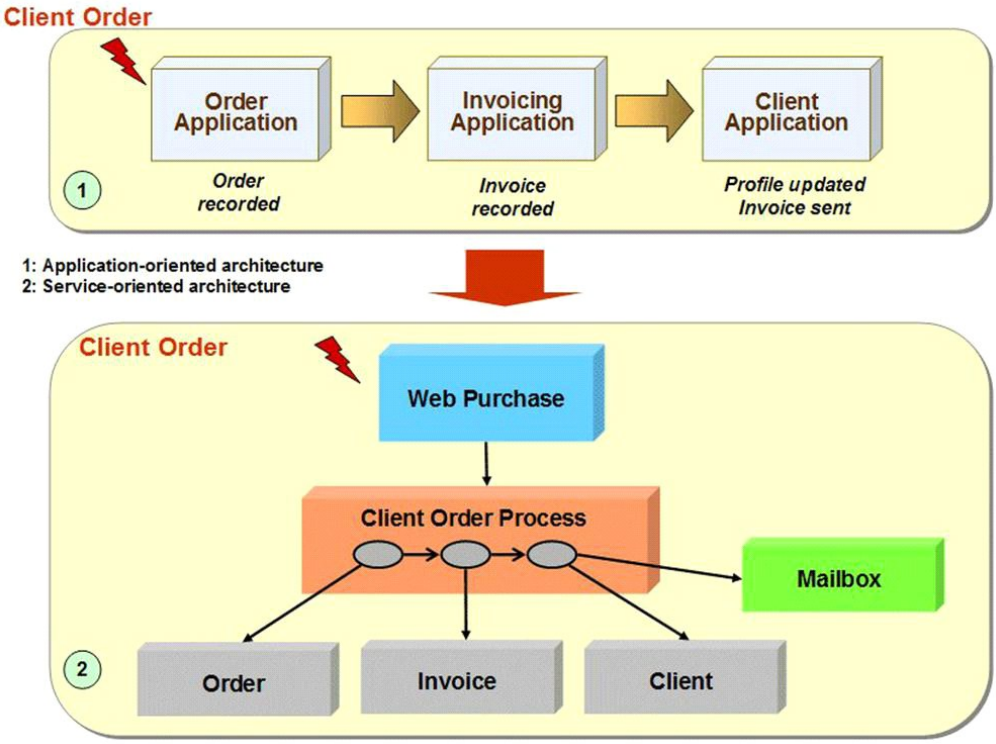

Figure 12.3: Application-oriented architecture versus the BPM/SOA approach

In the first case (1), each of the three applications is in charge of running a part of the process and then "hands over" to the next. In the second case (2), the process is totally handled by a dedicated component, with one major consequence: *the process itself exists in the system*, while in the previous case it was invisible, masked by the applications. The advantages of this architecture are twofold:

- Evolutions to the process are simpler to implement. The addition of a task or the adjustment of an execution path is handled in a centralized manner, in a framework that is perfectly adapted to this type of operation.
- Process supervision is greatly facilitated. BAM (business activity monitoring) tools fit naturally into a process execution engine, and provide precious assistance (execution reports, logs) in maintenance and improvement work.

From a BPM/SOA perspective, the best practice consists of clearly separating roles: the component in charge of running the process exclusively handles the sequencing and the rules that are directly associated with it. The other functions are delegated to services, such as data access or business processing. This structure, although ideal, is a particularly efficient "pattern," which can be adapted depending on what already exists, for example, by setting up "service mode" interfaces on applications without deploying an architecture that is 100% SOA.

[a]SOA is an approach to designing software that dissolves business applications into separate "services" that can be used independently of the applications of which they're a part and computing platforms on which they run (Jay DiMare, IBM Global Services, 2006).

[b]"SOA Reference Architecture"—see www.opengroup.org/subjectareas/soa. The work carried out by this working group was integrated into version 9.1 of TOGAF, published in December 2011 (chapter 22).

[c]"JBOWS (Just a Bunch of Web Services). An effective, functioning service-oriented architecture requires governance, and the ability to share services across multiple business units and enterprises. It's easy to build Web services. You could build 10 of them in an afternoon. But, then you end up with a JBOWS architecture (Just a Bunch of Web Services), which will grow into a different sort of SOA—a Spaghetti-Oriented Architecture," Joe McKendrick, Seven areas of opportunity around SOA, circa 2007, www.thegreylines.net/2006_12_01_archive.html.

[d]SOA: Architecture Logique: Principes, structures et bonnes pratiques, Gilbert Raymond, Softeam 2007, 2011, www.softeam.fr.

[e]TOGAF9 13.4.4.2.

[f]In actual fact, BPM covers a wider area, from modeling to supervision, but with the aim of increasing business process integration into IT systems.

12.2 BUSINESS PROCESSES

12.2.1 The Central Role of Business Processes

The day-to-day activity of an enterprise mainly consists of a set of employees participating in the running of business processes. Process management (BPM, Business Process Management) is a key domain, for which a whole series of approaches and techniques have been developed over several years: from process optimization with Six Sigma to the processes and process automation approach. Process management is a critical issue for all actors, CIOs, business owners, and project managers, in terms of system reactivity, activity monitoring, and market positioning.

As representatives of enterprise activity, business processes contribute greatly to the structuring of its architecture. When business processes evolve, architecture change projects are often initiated, with significant consequences for the information system. Moreover, the architecture repository often includes a set of process descriptions, from general mapping to detailed modeling, as a major element in the understanding of the inside functioning of the enterprise.

In the context of the ADM approach, processes mainly participate during phases A (Vision) and B (Business). Along with data entities, they form the core of business architecture. Because of their structuring nature, they also have a significant influence on the development of the information system and the technical foundations. The automation of certain processes accentuates their integration with the system: from the status of descriptive models, they find themselves at the heart of its functioning.

Given the importance of the subject, its contours, difficulties, and pitfalls must be defined. Later in this chapter, we will see that the very definition of a business process can vary, and that there exist a wide variety of types of processes within the enterprise. Add to this the fact that managing processes and their representations is anything but straightforward; the risk of getting lost in the sheer volume of information linked to processes is very real.

12.2.2 What is a Business Process?

This is the first question to ask, even if it seems trivial. All too often, the term "process" is used as a "catch-all," covering very different realities. If you need any convincing of this, just ask each person in a group to give his or her definition of the term "process" and let the debate take place. The diversity of answers is surprising, ranging from the description of screen sequences to the functions of the enterprise, or even the algorithms of IT processing (Figure 12.4).

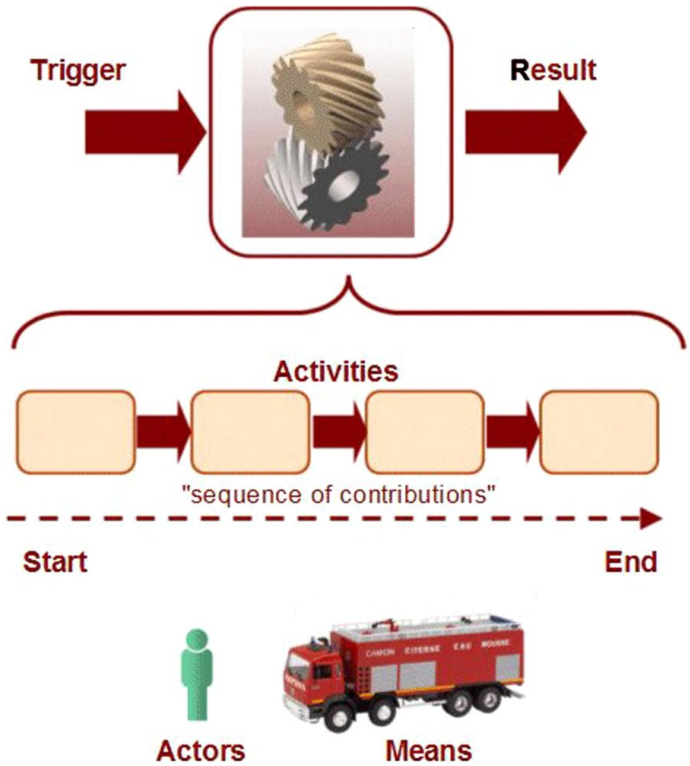

Figure 12.4: Business process, trigger, result, actors, means

TOGAF provides the following definition: "A process represents a sequence of activities that together achieve a specified outcome, can be decomposed into sub-processes, and can show operation of a function or service (at next level of detail). Processes may also be used to link or compose organizations, functions, services, and processes."

Fundamentally, a business process is a correlated set of activities producing tangible added value from an initial request (the trigger event). Activities are carried out by actors (human or automatic) using adapted means. The *business* nature of the process is expressed through the nature of the result, which must be meaningful for a client (internal or external) and measurable, where possible.

Here, the term "activity" designates a work unit that brings added value through the transformation or production of information or material. Other terms are also used, such as "task" or "stage."

12.2.3 Main Characteristics of Business Processes

Beyond the initial definition, certain characteristics enable the nature of business processes to be more precisely defined: cross-organization, temporality, parallelism, and event processing.

Cross-Organization

A process is inherently *cross-organizational* to the enterprise's functions and entities. It is made up of the different stages that follow one another in a more or less complex manner, from the trigger event until the final result is obtained. We talk about "end to end" processes, with key performance indicators (KPIs) that measure the quality of the final result. Figure 12.5 shows the progress of an order process, which runs across several enterprise entities (sales, invoicing, production, and delivery). Evaluation of this kind of process measures, for example, the time between an order being taken and the actual delivery of the product. Improvement in the quality of the service rendered to the client means working on the entire process (Figure 12.5).

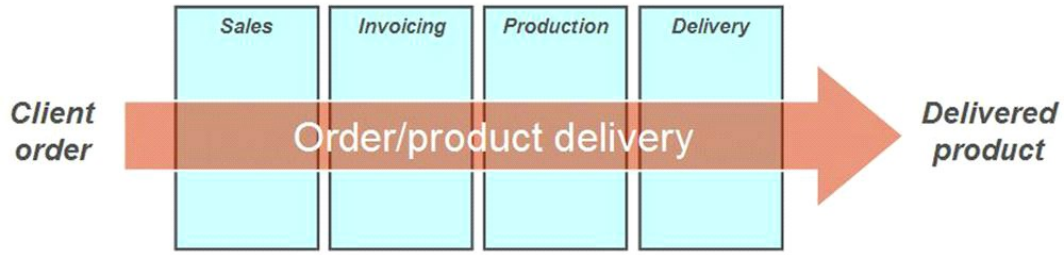

Figure 12.5: Process cross-organization

Depending on the complexity of processes, it may be necessary to perform a further breakdown into subprocesses. Some of these subprocesses can be used in several processes, which facilitates simplification through factorization.

Temporality

A process has a beginning and an end, and runs over a given period of time.[g] This statement may appear trite but is worth remembering. For example, "contract management" generally does not represent a process of the enterprise but rather a function or logical group of processes, since "contract management" has no start and no finish. The "contract management" group can include, for example, the "open a new contract" and "amend a contract" processes, which run over a given period of time between the moment they start and the moment the result is obtained.

Parallelism

When a process is run, we often find activities that run simultaneously, notably those that are undertaken by different actors. These parallel branches require that information be exchanged or synchronized. In the previous example, the enterprise can decide to run the invoicing and production activities at the same time. The actual delivery of the product waits for the end of these two activities before starting. This new organization reduces the overall duration of process execution, but requires strict synchronization of the end of the two invoicing and production activities. This is a typical choice for process architecture, with greater or lesser repercussions on the other architectural constituents involved, such as business organization or IT elements.

Events

Business processes are rarely isolated. They react to outside events that have a direct influence on their progress. In particular, certain activities will find themselves in the position of waiting for a given event to happen: the process is suspended until this event occurs. For example, when an insurance claim is processed, the investigation will await the results of the evaluation report before continuing. Other events can also interrupt an activity that is underway, such as a cancelation request.

Note that this event aspect is particularly present in BPMN notation, which provides no less than 50 different types of events.

12.2.4 Process Typology

In the context of business process descriptions, we are always confronted with the diversity of the situations encountered. This diversity is the expression of the reality of enterprise processes, which exist independently of their representation. Classification of the processes themselves and their internal functioning facilitates their management, both as a qualification tool and as an architecture repository organization tool (business process models are part of business architecture in TOGAF).

Generic Typology of Business Processes

The following classification is fairly widely used,[h] since it provides initial positioning of processes within the enterprise:

- *Operational processes or "core business" processes*, responsible for direct enterprise added value (claims processing, client orders, etc.).
- *Support processes*, which assist operational processes without directly participating in the results (price update, production and update of

product catalogs, etc.).

- *Management processes*, linked to the strategies and general management of the enterprise (market studies, general goal definition, supervisions, etc.).

This list can be completed by adding the "internal process" type, which designates processes that are not specific to the business in question, such as recruitment processes or resource management processes.

This typology facilitates the hierarchical organization of processes and enables better definition of their relationships. Operational processes are obviously the most critical for the enterprise, but each process contributes to obtaining defined goals.

Types of Sequences

The types of relationships that exist between the activities of a process are variable and define its functioning mode. The sequencing of activities can be strictly determined, or can be left more or less flexible. In the first case, we will talk about deterministic or "mechanistic" processes. Anyone looking at the example in Figure 12.6 will quickly come to the conclusion that the sequence of activities of the process in question cannot be different (unless we consider that it is normal to start by ironing and to finish by washing clothes). Different paths can exist, but these are strictly marked out by explicit conditions. In our example, ironing is an option depending on the client's wishes.

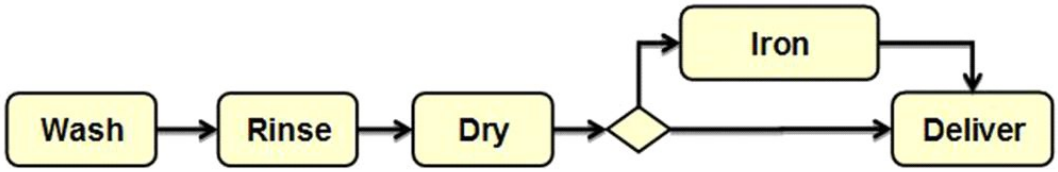

Figure 12.6: Simple example of a "deterministic" or "mechanistic" process

In the second case, the sequence of activities is much more random and can be the result of choices made by participating actors (Figure 12.7). This is notably the case with design or diagnosis processes, in which the path to the expected result cannot be described simply, even if the activities themselves are clearly defined.[i]

Figure 12.7: Nondeterministic process: Example of a medical diagnosis

In the example of the medical diagnosis, we can simplify by stating that each consultation or analysis activity can potentially redirect the patient to one or several other activities according to the practitioner's opinion. The exhaustive description of all possible paths would result in a highly complex, totally unusable graph.

We find a similar distinction in the workflow community between the "procedural" workflow and the "ad hoc" workflow. Procedural workflows (also called production workflows or directive workflows) are business processes known to the enterprise, which are the subject of preestablished procedures, and whose sequence of activities is fixed. Ad hoc workflows are based on a collaborative model in which actors participate in the decisions regarding their sequence of activities.

Knowing which type of sequence of activities is involved (even without going into detail at this level) conditions certain decisions, notably with regard to modeling work and implementation choices.

Most modeling languages such as BPMN are better adapted to describing mechanistic processes, with a fixed sequence of activities.[j] Some approaches advocate representation through event and condition tables in the case of nondeterministic processes.

It is clear that process automation implementation can vary significantly depending on the type of sequence of activities. Once again, process execution history tools facilitate deterministic processes, but the emergence of "case management" solutions[k] provides a more collaborative vision of processes.

Batch Processing and Desynchronization

We have seen that a process takes place through the sequence of its activities. Activities represent work units, each of which contributes to the final result, as in an industrial production line. Let's take a typical example of a (highly simplified) process concerning order processing. This process (which is deterministic) is broken down into three activities: validation, invoicing, and order delivery (Figure 12.8). It is triggered when the order is received.

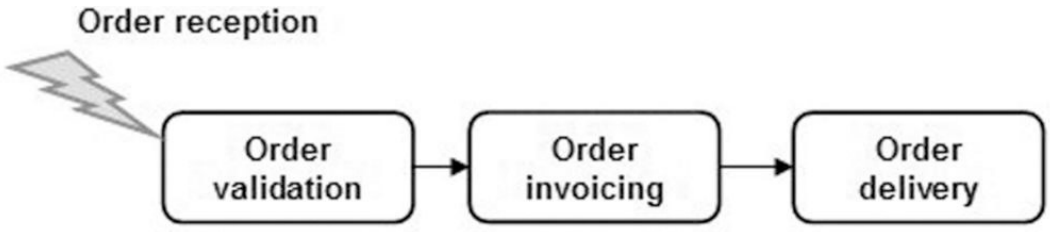

Figure 12.8: Simple example of an order process

But if, as is often the case, the enterprise decides to group several orders into a single delivery that takes place at the beginning of every week, this description becomes incorrect. The "Order delivery" activity does not exist. Furthermore, the real activity of delivery of a batch of orders is not triggered following order invoicing, but rather at a fixed date (at the beginning of the week). A more realistic representation of the process is provided in Figure 12.9.

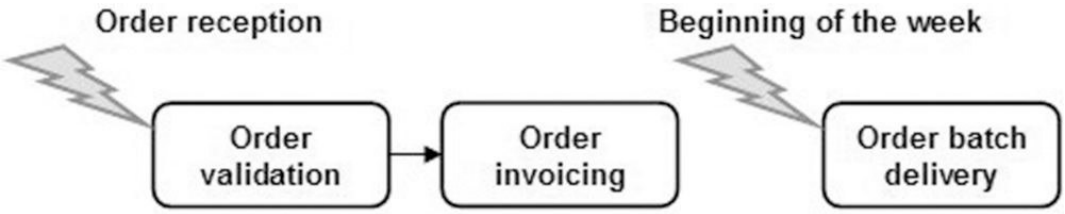

Figure 12.9: Example of desynchronization through batch processing

The process ends up being "decoupled" into two separate processes, each with a different trigger. This is a recurring question with regard to processes and work organization methods: the choice between "just in time" processing and "batch" processing. According to the context, the enterprise can choose to manage each order individually right through to delivery, or as we have just seen to deliver batches of orders.[1]

It is easy to imagine that the consequences of this choice on the organization and architecture will be significant: an accumulation of orders over the week, product storage issues, delivery implementation, and so on. Thus, it is essential that desynchronization phenomena be identified in order to properly carry out work on the resulting architecture.

12.2.5 Describing and Modeling Business Processes

Process Modeling: A Risky Business?

All too often, rushing into detailed modeling of processes leads to a paradoxical situation: the proliferation of models that are difficult to use and quickly obsolete. Many enterprises that have undertaken the large-scale modeling of their processes observe that results are poor compared with the investments made: the reality of the situation has evolved while a significant part of the repository has been "treading water."

Note that here we are talking about process models that are intended to last, like architecture repository elements. Descriptions, models, or various representations realized for temporary needs are not concerned. By nature, these are "perishable" and are not subject to the same durability constraints.

Of course, this disconnection between reality and description can be seen for each part of the architecture repository. However, the risk of a break with reality is particularly high for business process models for two essential reasons: first, the elevated cost of investment in process modeling and, second, the frequently changing nature of the subject being dealt with. Modeling a process consists in "dissecting" activities, with all the implicit imprecision and knowledge this includes. The result will only be convincing through the involvement of business resources, which are sometimes difficult to mobilize within a reasonable timeframe. Consequently, the representation of real processes goes beyond simplistic diagrams made up of "boxes" and "arrows," and requires real introspection on the intimate functioning of the enterprise.

How should one proceed faced with this situation? Above all, by starting from the goal in terms of communication and use, and by adapting the form and content of the representation in accordance with this goal: to represent in the repository what we really need and what needs to be maintained over time.

Identification, Qualification, and Modeling

With this in mind, we suggest the definition of three distinct levels of representation for business processes: identification, qualification, and modeling.

This distinction has two advantages. First, it organizes the process repository into successive layers, by increasing the level of detail, and second, it encourages a more progressive approach, which tends to be quickly beneficial by minimizing investment (both cost and time).

Identification

The *identification* of processes can take several different forms: a simple inventory or an organized mapping. For each process, a set of fundamental information is grouped in the *process identity file*: the trigger event, input and output, participants resources used, key

performance indicators (KPIs), and so on. Easily accessible and concise, this file is the entry point to knowledge of the process. The following table gives an example of a process identity file for the order process.

Property	Description
Finality	Delivery of products ordered by the client within set time limits
Trigger event	Receipt of the client's order
Input	Order form
Output	Invoice, product
Key performance indicators (KPIs)	Total duration of the process <3 days
Responsible actor (governance)	Process driver has been designated
Resources used	CRM, delivery management application
Main actors	Client, order manager, delivery service
Work in progress	Optimization study

The Six Sigma method[m] provides a similar tool with the SIPOC (*Supplier, Input, Process Output, Customer*), which macroscopically presents the process in the form of a table or a diagram. TOGAF uses the ICOM acronym: *Inputs, Controls, Outputs, and Mechanisms/resources used*.

Process identification can be widened through the incorporation of *process mapping*. This provides an overview of the enterprise, which enables each process to be positioned in a predefined context (based on business domains, for example). Process mapping positions each process in relation to other processes, and defines the main relationships in the form of interprocess exchanges.[n]

Qualification

The goal of *qualification* is to better define processes in all their diversity and to facilitate decision making. The characteristics grid enables each process to be defined in the same way, using criteria that can be directly used: frequency, complexity, duration, reported malfunction, typology, number of participants, and so on.

> **Note** It is not possible to compare a totally automated process that is run 200 times per minute, and another that is managed entirely by human actors and that is run twice a year. The constraints involved, the skills used, and the range of possible solutions vary greatly from one situation to another.

This qualification sometimes requires more in-depth study of the process, for example, by identifying the typologies previously described. Depending on the situation, the use of Six Sigma or Lean-type methods will be well suited, such as statistical measurement of errors, blockages, or resource consumption[o] for optimization purposes.

Beyond the information it provides, the result of qualification will guide certain architectural choices, primarily the decision as to whether or not to automate the process, and if so, in what form. This point is discussed later in this chapter.

Modeling

Models based on graphical notation are essential to the detailed representation of business processes, as with the BPMN[p] standard.

Figure 12.10 presents a simple example of a business process described with BPMN. The "Order reception" trigger event starts up the process, which begins with validation of the order. Two parallel branches are then run: dispatch and delivery confirmation on the one hand, and invoicing and payment on the other. The process only ends when both these branches have been completed.

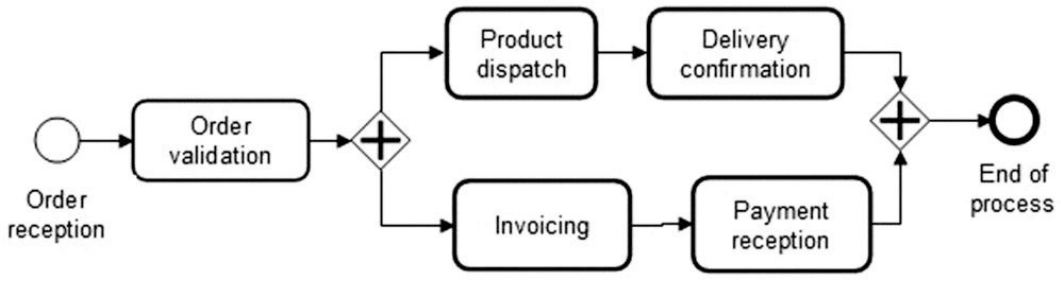

Figure 12.10: Example of a BPMN diagram

Models have long been used in many sectors to understand, develop, simulate, and communicate (see Chapter 5). Modeling requires particular skills and knowledge: choice of level of detail, gathering and consolidation of information, and communication of results. The type of modeling carried out will vary depending on the goal (general description, detailed information on the process, or support for automation).

Using this principle, Bruce Silver[q] identifies three levels of business process model: the *descriptive* level, the *analytical* level, and the *executable* level. The descriptive level provides the fundamental structure of the process, with its main activities, but without taking exceptions

into account. The analytical level establishes the sequence of the process in detail, with all its activities and exchanges. The executable level is used in the context of process automation by integrating links to software elements and technical constraints. Once again, we observe the need to separate viewpoints applied to modeling.

Process Description Management

Using TOGAF terminology, the breakdown into levels that we have just discussed consists of defining a set of viewpoints dedicated to processes:

- The "identification" viewpoint, which presents high-level views that are simple to use but which provide a summary of the main process-related information.
- The "qualification" viewpoint, which is the result of analysis of the process, its characteristics, and possible improvements.
- The "modeling" viewpoint, which describes process content in a more or less detailed way, using BPMN-type notation.

This structuring in the architecture repository enables more efficient communication and reduces the risk of uncontrolled proliferation of process descriptions.

In practice, the aim is not to define all the enterprise's processes in one "big bang" operation. Priority will be given to "core business" processes or opportunity-driven processes (optimizations, evolutions). Generally speaking, the number of "identified" processes is greater than the number of "qualified" and "modeled" processes. We strongly recommend that the switch from one level to another be clearly justified, notably for modeling, in view of the investment necessary for this type of work.

12.2.6 Process Governance

The Process Driver

The "process driver"[r] function is used in many enterprises to guarantee the quality, monitoring, and continual improvement of business processes. Today, a whole host of publications exist on the subject, as well as sharing communities such as the "Process Driver Club."[s] It has multiple roles and touches on several facets:

- Description and modeling
- Improvement and optimization
- Coordination of the application components involved in the process
- Training and informing of users
- Supervision of execution, key performance indicators
- Feedback from monitoring and information/warnings provided to management
- Maintenance, support, and parameterization
- Study of changes
- Repository update

Furthermore, this function emerged relatively late and is still having difficulty in widely establishing itself. This is paradoxical since everyone is convinced of the critical role played by processes within the enterprise.

The main reason for this delay is the cross-organizational nature of business processes. We have already seen that business processes generally run over several functions, managed by distinct organization units. This positioning goes against the historical and sometimes "hierarchical" structure of enterprises. "Traditional" IT applications easily find their place in this type of organization, under the clearly defined responsibility of well-established units. However, installing and stabilizing a cross-organizational-type organization, which plays a real driving role, is a delicate task. That said, this depends on the business context. In fields closely linked to logistics, such as transportation, processes are naturally seen as being at the heart of the business. For other types of activity, the approach by process has only really begun to develop over the past few years.

Role in Enterprise Architecture

Enterprise architecture, another cross-organizational activity, is an opportunity for process drivers, which are very often found in the frontline during architecture development and transformation work. In this respect, process drivers have to participate in transformation work, throughout all phases of the ADM cycle. In some cases, it is the creation of an architecture project that triggers the implementation of specific process management, which continues beyond the ADM cycle.

12.2.7 BPM, BPMN, Standards, and Tools

In the context of BPM work, several types of tools are available. In its studies, Gartner® distinguishes BPA (*Business Process Analysis*) tools

and BPMS (*Business Process Management Suite*) tools. BPA tools are dedicated to process repository modeling and management, while BPMS tools are positioned as automated process development and deployment platforms. Furthermore, tool publishers are tending to move toward BPMN, which meets the increasing need for a normalized representation language, for both BPA and BPMS tools. BPMN support is available in most UML tools in order to maintain links to other models (such as the class diagram, for example). This proximity of the two standards lends even more weight to BPMN in the context of tool compatibility.

However, the range of process modeling tools is more widely diversified in enterprises. The use of graphical tools (Visio®, PowerPoint®) or office solutions is widespread. This is fine for initial descriptions, which require easy, flexible usage, but quickly turns out to be counterproductive when managing a structured whole over time.

Figure 12.11 shows the evolution of standards linked to BPMN. Three types of standard are represented: modeling languages from which BPMN has resulted, exchange formats, and execution languages (BPEL).

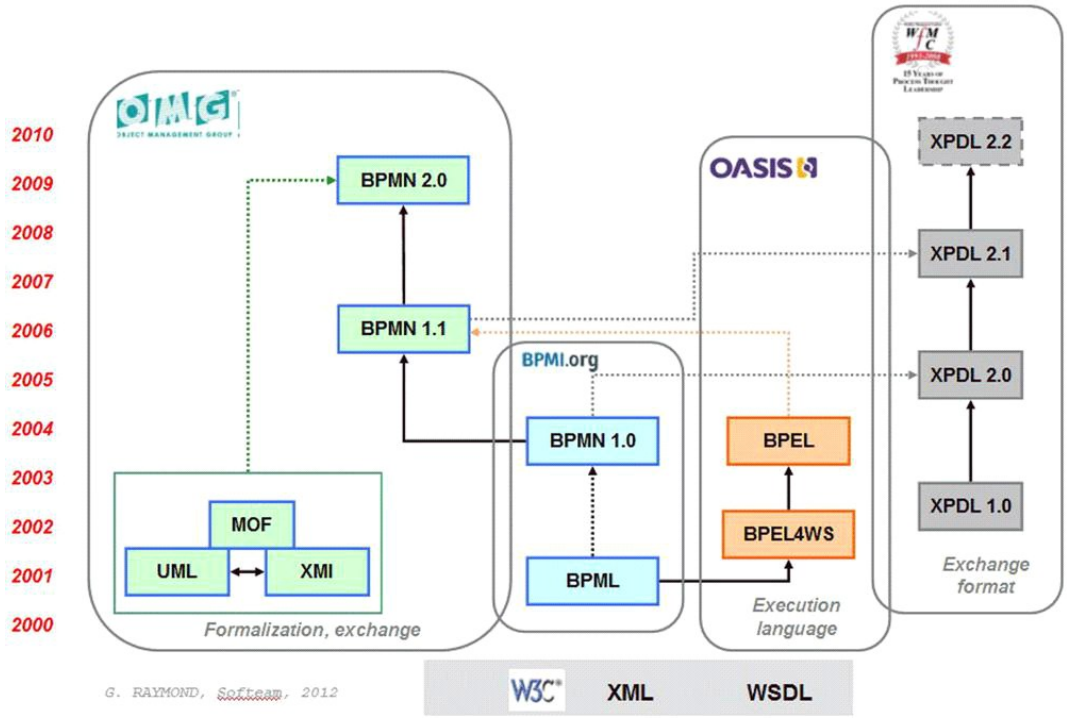

Figure 12.11: History of standards linked to BPMN

Historically, BPMN was first built as a graphical process notation by the BPMI.org[t] consortium. This consortium merged with the OMG in 2005, resulting in recent versions (2.0) in a more rigorous and formal formulation based on the MOF, XMI, and UML techniques for the definition of the metamodel and the exchange format. The WFMC (WorkFlow Management Coalition) developed XPDL, a process exchange format based on XML and compatible with BPMN. BPEL plays a special role: it is a language and file format that is managed by a process execution engine.

Even if the BPMN standard defines BPEL mapping rules, the generation of BPEL files from BPMN process models remains a delicate operation,[u],[v] BPEL is one way to execute processes, but some runtime platforms can execute BPMN directly, thereby eliminating the need to carry out a transformation.

[g]This period of time can be long (years), referred to as a "long-term" process.

[h]Typology from the ISO 9001 norms.

[i]For this type of process, we also talk about "case management."

[j]To be exact, BPMN contains the "ad hoc" type of activity to respond to this kind of process. However, this element is little known and little used.

[k]Case Management: A Review of Modeling Approaches, Henk de Man, BP Trends, January 2009.

[l]Yves Caseau discusses this issue in his book, specifying that the interpretation of process diagrams varies depending on element semantics: the transition arrow between two tasks can represent a strict sequence as in BMPN (these are the semantics implied here) or simply an existing relationship. Urbanisation, SOA et BPM—Le point de vue du DSI, Yves Caseau, Dunod, 2008.

[m]Lean Six Sigma: Overview of techniques, http://itil.fr/LEAN-SIX-SIGMA/lean-six-sigma-tour-dhorizon. html.

[n]An example of a process mapping model is presented in Chapter 7.

[o]It is not a question of systematically rolling out the Six Sigma or Lean methods, but rather of using certain tools according to different needs. Intense optimization work is outside the scope of this chapter.

[p]Business Process Modeling Language, standard managed by the OMG consortium (omg.org).

[q]BPMN Method and Style: A Levels-Based Methodology for BPM Process Modeling and Improvement Using BPMN 2.0, Bruce Silver, Cody-Cassidy Press, 2009.

[r]Or "process owner."

[s]www.pilotesdeprocessus.org.

[t]http://www.bpmi.org.

[u]Translating BPMN to BPEL, Chun Ouyang, Wil M.P. van der Aalst, Marlon Dumas, and Arthur H.M. ter Hofstede, 2006, http://eprints.qut.edu.au.

[v]Why BPEL is not the holy grail of the BPM, Pierre Vignéras, 2008, Bull SAS.

12.3 INFORMATION

If business processes are at the heart of an enterprise's activity, information constitutes an equally fundamental part. Information systems are primarily built as tools for processing information, and guaranteeing the quality of data remains one of their most critical goals.

Databases are the first image that comes to mind when discussing enterprise information systems. While databases do play a central role, can we really say that they are a unique source of information?

In the different activities related to enterprise architecture, we also need to consider the different forms of information handled within a complex organization.

12.3.1 Different Types of Information in the Enterprise

Structured and Nonstructured Information

In general, we can identify two main categories of information that coexist within an enterprise: structured information and nonstructured information.

- *Structured information* is organized according to a preconceived, defined archetype. Each element of information corresponds to an element that specifies its type and its value domain. These information elements can be handled directly through IT processing. Databases constitute the main support for this type of information.
- *Nonstructured information*, such as textual documents, does not follow a predefined format. It is organized (for example, into chapters), but particular tools are required to process it. This is the field of electronic content management (ECM).

In recent times, two opposing movements have emerged. On the one hand, the transfer of a large amount of nonstructured information to a structured formulation: an order previously transmitted via a paper order form is now directly entered into a form via a website. On the other hand, the multiplication of nonstructured information in all its different forms: electronic messages, videos, forums, and so on. Let's not forget that the number of documents available within an organization is often considerable, and that these documents contain a large proportion of its know-how.[w]

Enterprises (and their employees) have opened up to the outside world, and this has further extended the boundaries of their communication. How can one not regard the web as a gigantic reservoir of nonstructured information?

Consequently, concentrating all efforts on structured information is too restrictive, and does not allow the reality of work modes today to be taken into account. Enterprise architecture also means looking into intranets, document organization, or the efficient management of electronic mail systems.[x]

Resources and Messages

Like stocks and flows, information is presented in two different ways:

- *Persistent information*, or resources, which last beyond business activities and processes.

- *Exchanged information*, in the form of messages whose content has a limited duration.

By definition, a client repository or a regulatory document contains persistent information. However, interapplication flows do not last. It is clear that persistent information requires special management in order to guarantee that it remains pertinent throughout its lifecycle. This is typically the field of the MDM (*Master Data Management*), which uses enterprise data synchronization and general management tools. However, the information transmitted during exchanges is more volatile and subject to fewer constraints. We will see that this can lead to some confusion.

Table 12.2 summarizes the different types of information that we have just discussed, with examples corresponding to the different possible cases.

Note The big issue is that people confuse the schemas for persistent data sources with the schemas for exchanged data, hoping there is some simple automated mapping between the two. There generally isn't and these need to be modeled more or less independently. This point is crucial and must be managed accordingly (see Section 12.3.2).

Table 12.2: Examples of Different Types of Information

	Persistent	Exchange
Structured	Database	Interapplication flows
Nonstructured	Documents, intranet	E-mail

12.3.2 Data Exchange in the System

Automatic data exchange is widely used in information systems. This can be synchronous exchanges between two components, or batch processing run at fixed times.

In all cases, automatic processing means that structured data must be used. However, persistent data and exchange data are organized in very different ways.

While persistent data is organized according to strict procedures (by respecting normal forms in relational databases), exchange data is organized in a much less regimented manner. Its content is above all driven by pragmatic, needs-driven considerations: a message[y] will convey the contract and the last 10 orders placed by a client, for example (Figure 12.12).

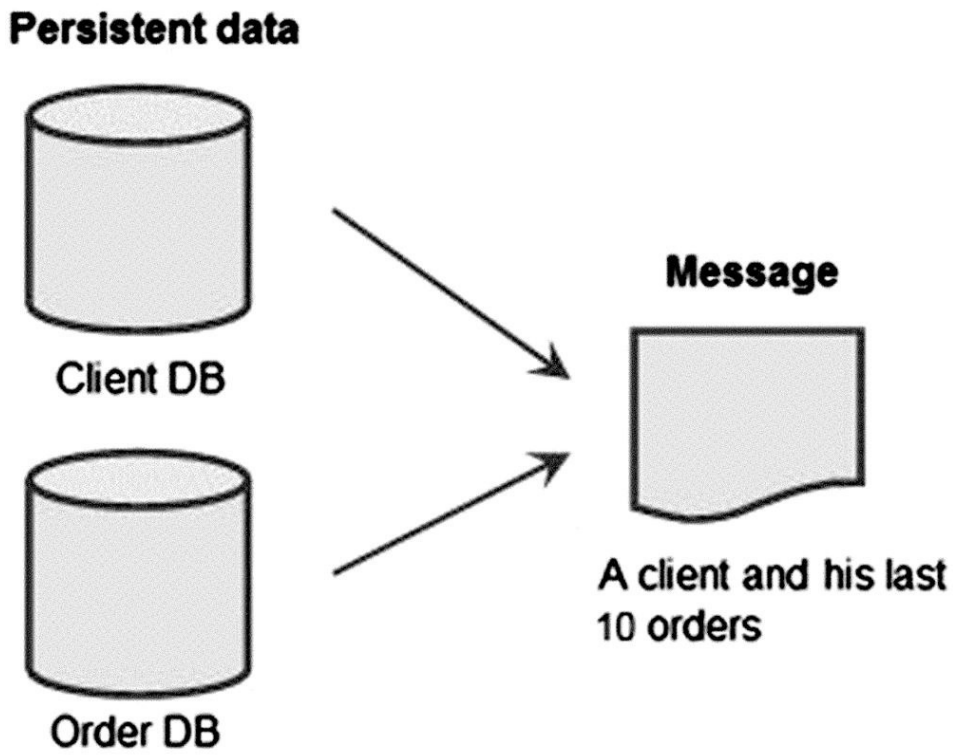

Figure 12.12: Persistent data and exchange data

These data exchanges have been present for a long time in information systems. This is the famous "spaghetti" that is presented as the worst possible thing in architecture. The negative nature of this culinary comparison mainly underlines the chaotic aspect when links proliferate between applications. The structure and content of information flows is progressively constituted from localized needs, with no real overall plan. The increase in exchanges goes hand in hand with the multiplication of messages, which in turn leads to an increase in complexity. A colleague recently told us that more than 6000 different types of messages were catalogued in his enterprise, some of which are used in the

most basic of ways.

Faced with this type of situation, some enterprises have chosen to undertake normalization and rationalization work through the centralized definition of messages (centralized or common format). The aim of this work is to reduce the number of duplicates (for example, the existence of several, virtually identical "contract" messages), and to facilitate connections between application elements. The use of technologies linked to web services facilitates this rationalization work, notably the use of XML documents to support exchanges; messages are specified using XML schemas,[z] which enable the structure of exchanged information to be automatically checked.

That said, there is a "natural" diversity in the messages exchanged. This diversity, which is well known, is the expression of the sheer number of viewpoints that coexist within an enterprise, just like the "client" type, which is dealt with in different ways by the marketing, invoicing, or delivery departments.

What is the relationship between persistent data and exchange data? In general, a message does not consist of a simple copy of persistent data. Its content can come from several items of data, either as the result of a conglomeration of a set of properties or as a value calculated from a set of properties.

Work on harmonizing and industrializing messages is confronted with two paradoxical stumbling blocks: on the one hand, imposing too much proximity between exchange data and persistent data (in other words, between messages and database tables) and, on the other hand, allowing uncontrolled structures to proliferate, resulting in unmanaged situations.

In any case, the search for a balance between these two extremes is one of the main aspects of enterprise architecture work.

12.3.3 Managing Interoperability

On this question, TOGAF provides a classification consisting of four degrees[aa]:

- Degree 1: Unstructured data exchange, entirely handled by human participants (exchange of documents)
- Degree 2: Structured data exchange, which requires some manual exchange operations (reception, distribution)
- Degree 3: Seamless data sharing, based on a shared exchange format
- Degree 4: Extension of degree 3: A set of cooperating applications based on a reference model

The choice of exchange mode is a key issue, with repercussions on both work organization and implemented applications.

In the context of a transformation project, earlier choices are regularly questioned, for example, to move toward increased automation of exchanges or to better centralize information that is spread across several repositories.

Here again, the choice of solution must take different factors into consideration (organization, IT, cost). Above all, the aim is to justify choices with regard to real business results. For example, is the replacement of a nonstructured document transmission by a structured data exchange really relevant?

It should be pointed out here that this question concerns exchanges in general, and not only exchanges between applications. In phase B notably, the question of exchanges between business actors arises. The following matrix[bb] presents an overview of the type of interaction between different users. Each cell qualifies the type of exchange between users using degrees from the aforementioned list (from 1 to 4) (Figure 12.13).

Phase B: Inter-stakeholder Information Interoperability Requirements (Using degrees of information interoperability)							
Stakeholders	A	B	C	D	E	F	G
A		2	3	2	3	3	3
B	2		3	2	3	2	2
C	3	3		2	2	2	3
D	2	2	2		3	3	3
E	4	4	2	3		3	3
F	4	4	2	3	3		2
G	2	2	3	3	3	3	

Figure 12.13: Business information interoperability matrix—TOGAF9

This type of representation will also be used for gap analysis, between the types of exchange in place and those that are envisaged in the new architecture.

[w]The TOGAF reference document is part of this nonstructured information.

[x]A study by the Radical Group shows that executives receive an average of 80 e-mails per day, directly impacting their productivity. A French enterprise is even considering withdrawing this type of exchange in the future.

[y]We also see the term business data type (BDT): Business Component Factory, Peter Herzum & Oliver Sims, Wiley Computing Publishing, 2000.

[z]An XML schema is used to specify the structure and content of an XML document.

[aa]TOGAF9 29.4 From the Canadian Department of National Defense and NATO.

[bb]TOGAF9, Figure 29-1.

12.4 FUNDAMENTAL CONCEPTS

The following fundamental concepts were introduced in this chapter:

- SOA: Style of architecture based on the concept of service, designed to simplify interactions between architecture blocks while providing the system with significant flexibility.
- Business process: Correlated set of activities that produces tangible added value from an initial request (the trigger event).
- Information types: Information can be classified into structured/nonstructured and persistent/exchanged.