

HOPEX IT Architecture

User Guide

HOPEX V3



Information in this document is subject to change and does not represent a commitment on the part of MEGA International.

No part of this document is to be reproduced, transmitted, stored in a retrieval system, or translated into any language in any form by any means, without the prior written permission of MEGA International.

© MEGA International, Paris, 1996 - 2019

All rights reserved.

HOPEX IT Architecture and HOPEX are registered trademarks of MEGA International.

Windows is a registered trademark of Microsoft Corporation.

The other trademarks mentioned in this document belong to their respective owners.

CONTENTS



| | |
|---------------------------|----------|
| Contents | 1 |
|---------------------------|----------|

| | |
|--|----------|
| Introduction to HOPEX IT Architecture | 9 |
|--|----------|

| | |
|--|-----------|
| Presentation of HOPEX IT Architecture | 11 |
|--|-----------|

| | |
|---|----|
| The Scope Covered by HOPEX IT Architecture | 11 |
| Summary of Activities and Deliverables of HOPEX IT Architecture | 12 |
| Structure and positioning of the HOPEX IT Architecture Solution | 12 |
| HOPEX IT Architecture Profiles | 13 |
| Business Roles of HOPEX IT Architecture | 15 |

| | |
|---|-----------|
| The HOPEX IT Architecture method | 16 |
|---|-----------|

| | |
|---|----|
| Building the Business Capability Maps of the Organization | 16 |
| <i>Describing Business Capabilities</i> | 16 |
| <i>Identifying the functionalities associated with business capabilities</i> | 17 |
| <i>Identifying the applications associated with business capabilities</i> | 19 |
| Building the Logical Architecture | 19 |
| <i>Structure diagram of the logical application system</i> | 21 |
| <i>Logical application system environment diagram</i> | 22 |
| Describing Application Architecture | 23 |
| <i>Describing application systems</i> | 24 |
| <i>Describing application processes</i> | 26 |
| <i>Describing flow scenarios</i> | 27 |
| Defining the Application Technical Architecture | 28 |
| Defining the infrastructure | 29 |
| <i>Resource Architecture Environment Diagram</i> | 29 |
| <i>Describing Resource Architectures</i> | 30 |
| <i>IT infrastructure assembly structure diagram</i> | 31 |
| <i>Computing Device Assembly Diagram</i> | 33 |
| Analyzing the Functional Coverage of the Architecture Implemented | 33 |
| <i>Analyzing the functional coverage of the application architecture</i> | 33 |
| <i>Analyzing the Functional Coverage of the Application and Hardware Architecture</i> | 34 |
| Managing Service Catalogs | 35 |

| | |
|--|-----------|
| HOPEX IT Architecture Desktop Presentation. | 37 |
| Connecting to the solution | 37 |
| HOPEX IT Architecture Desktop Presentation | 37 |
| <i>Presentation of space common to all profiles</i> | <i>38</i> |
| <i>Presentation of the IT Architecture Functional Administrator space.</i> | <i>39</i> |
| <i>Presentation of the IT Architecture Manager space</i> | <i>42</i> |
| <i>Presentation of the Application Architect space</i> | <i>42</i> |
| <i>Presentation of the Infrastructure Architect space</i> | <i>43</i> |
| <i>Presentation of the IT Technical Architect space.</i> | <i>44</i> |
| <i>Presentation of the Application Contributor space.</i> | <i>45</i> |
| <i>Presentation of the Application Viewer space</i> | <i>45</i> |
| Switching Between Profiles. | 46 |
| Before starting with HOPEX IT Architecture. | 47 |
| Preparing the Work Environment | 47 |
| <i>Accessing the list of libraries with HOPEX IT Architecture</i> | <i>47</i> |
| <i>Creating a library with HOPEX IT Architecture</i> | <i>47</i> |
| <i>Accessing the list of enterprises with HOPEX IT Architecture</i> | <i>48</i> |
| <i>Creating an enterprise with HOPEX IT Architecture.</i> | <i>48</i> |
| <i>Choosing a Working Environment with HOPEX IT Architecture.</i> | <i>48</i> |
| <i>Switching Between Profiles with HOPEX IT Architecture.</i> | <i>48</i> |
| Using properties pages | 49 |
| <i>Displaying the properties window on a permanent basis</i> | <i>49</i> |
| <i>HOPEX IT Architecture properties pages content</i> | <i>49</i> |
| Importing components with HOPEX IT Architecture | 51 |
| <i>Structure of the import/export Excel templates of HOPEX IT Architecture</i> | <i>52</i> |
| <i>Importing computing devices or technologies with Excel.</i> | <i>53</i> |
| <i>Building the import file for HOPEX IT Architecture</i> | <i>55</i> |
| About This Guide. | 59 |
| Guide Structure | 59 |
| Additional Resources | 59 |
| Conventions used in the guide | 60 |

Modeling Business Capabilities and the logical application architecture 61

| | |
|--|-----------|
| Using Business Capabilities with HOPEX IT Architecture | 62 |
| Business capabilities examples with HOPEX IT Architecture | 62 |
| Managing the Business Capability Maps with HOPEX IT Architecture | 63 |
| <i>Accessing the list of business capability maps</i> | <i>63</i> |
| <i>Creating a business capability map.</i> | <i>63</i> |
| <i>Creating a business capability map diagram</i> | <i>63</i> |
| <i>The properties of a business capability map.</i> | <i>64</i> |
| Managing Business Capabilities with HOPEX IT Architecture | 64 |
| <i>Describing a business capability</i> | <i>64</i> |
| <i>Accessing the list of business capabilities with HOPEX IT Architecture.</i> | <i>65</i> |
| <i>Defining the functionalities associated with business capabilities</i> | <i>65</i> |
| Describing implementation of a business capability | 65 |
| <i>Creating a business capability realization.</i> | <i>65</i> |
| <i>Analyzing enterprise capability implementation</i> | <i>66</i> |

| | |
|---|-----------|
| Using Functionalities with HOPEX IT Architecture | 67 |
| Describing a Functionality Map with HOPEX IT Architecture | 67 |
| <i>Accessing the list of functionality maps with HOPEX IT Architecture</i> | 67 |
| <i>The properties of a functionality map</i> | 68 |
| <i>Creating a functionality map</i> | 68 |
| Describing functionalities with HOPEX IT Architecture | 68 |
| <i>Creating a Functionality Diagram with HOPEX IT Architecture</i> | 69 |
| Describing Implementation of a Functionality | 69 |
| <i>Creating a Functionality Realization</i> | 69 |
| <i>Analyzing functionality implementation</i> | 70 |
| Describing a Logical Application Architecture with HOPEX IT Architecture | 71 |
| Describing a Logical Application System with HOPEX IT Architecture | 71 |
| <i>Accessing the list of logical application systems with HOPEX IT Architecture</i> | 71 |
| <i>Creating a Logical Application System</i> | 71 |
| <i>Logical Application System Properties</i> | 72 |
| <i>Describing a Logical Application System with HOPEX IT Architecture</i> | 72 |
| Describing Logical Applications With HOPEX IT Architecture | 75 |
| <i>Accessing the list of logical applications with HOPEX IT Architecture</i> | 75 |
| <i>Creating a logical application</i> | 75 |
| <i>Logical Application Properties</i> | 75 |
| Logical Application System Environment Description | 76 |
| <i>Example of logical application system environment</i> | 76 |
| <i>Accessing the list of logical application system environments</i> | 77 |
| <i>Creating a logical application system environment</i> | 77 |
| <i>Logical application system environment properties</i> | 77 |
| <i>Using the Environment Structure Diagram of a Logical Application System</i> | 77 |
| Using Org-Units in a Diagram | 78 |
| <i>Creating an org-unit</i> | 78 |
| <i>Internal org-unit/external entity</i> | 79 |

Modeling Technical and Functional Architectures81

| | |
|--|-----------|
| Describing Application Architecture | 82 |
| HOPEX IT Architecture Concepts Overview | 82 |
| <i>Application</i> | 82 |
| <i>Application System</i> | 82 |
| Describing an Application with HOPEX IT Architecture | 83 |
| <i>Creating an Application with HOPEX IT Architecture</i> | 83 |
| <i>The properties of an application with HOPEX IT Architecture</i> | 84 |
| <i>Defining Application Functional Scope</i> | 85 |
| <i>Specifying the Risks Associated with an Application</i> | 85 |
| Describing an Application Environment with HOPEX IT Architecture | 86 |
| <i>Accessing the List of Application Environments</i> | 86 |
| <i>Creating an application environment</i> | 86 |
| <i>Application environment properties</i> | 86 |
| Describing an Application System | 86 |
| <i>Creating an Application System</i> | 87 |
| <i>Application System Properties</i> | 87 |
| <i>Creating an application system structure diagram</i> | 88 |

| | |
|---|------------|
| Describing an Application System Environment with HOPEX IT Architecture | 90 |
| <i>Accessing the list of application system environments.</i> | 90 |
| <i>Creating an application system environment</i> | 90 |
| <i>Application system environment properties</i> | 90 |
| <i>Describing an application system environment diagram</i> | 90 |
| Describing an IT Service with HOPEX IT Architecture. | 92 |
| <i>Accessing the list of IT services</i> | 92 |
| <i>IT Service properties.</i> | 92 |
| Describing a Micro-Service with HOPEX IT Architecture | 93 |
| <i>Accessing the list of micro-services.</i> | 93 |
| <i>Micro-Service properties with HOPEX IT Architecture</i> | 93 |
| <i>Using the IT Service and Micro-Service Structure Diagram</i> | 93 |
| Describing Application Processes. | 94 |
| <i>Accessing the list of application processes</i> | 94 |
| <i>Creating a system process diagram</i> | 94 |
| Managing Data | 96 |
| Using Data Stores. | 96 |
| <i>Introduction to data store concept</i> | 96 |
| <i>Accessing to data areas with HOPEX IT Architecture</i> | 96 |
| <i>Creating a data area.</i> | 96 |
| Using Data Stores. | 97 |
| <i>Introduction to the data store concept</i> | 97 |
| <i>Usage contexts</i> | 97 |
| <i>Creating a local data store.</i> | 98 |
| <i>Creating an external data store</i> | 98 |
| <i>Describing access to a data store</i> | 99 |
| Describing Application Flows | 100 |
| Creating a Global Application Flow Map | 100 |
| <i>Accessing the list of global application flow maps</i> | 100 |
| <i>The properties of a global application flow map</i> | 100 |
| Using a Scenario of Application System Flows | 100 |
| <i>Adding a scenario of application system flows</i> | 101 |
| <i>Adding an org-unit to the Scenario of Application System Flows.</i> | 102 |
| Using a Scenario of IT Service Flows and Micro-Service Flow | 102 |
| Describing a Scenario of Application System Environment | 103 |
| Describing Technical Architecture | 105 |
| Describing a Software Technology. | 105 |
| <i>Accessing the list of software technologies</i> | 105 |
| <i>The properties of a technology.</i> | 105 |
| Describing Software Technology Stacks. | 105 |
| <i>Accessing the list of technology stacks</i> | 106 |
| <i>Properties of a technology stack.</i> | 106 |
| Creating an Application Technical Architecture | 106 |
| <i>Accessing the application technical architectures</i> | 107 |
| <i>Properties of an application technical architecture</i> | 107 |

Modeling IT Infrastructures 111
Describing a Resource Architecture 112

Creating a Resource Architecture 112

Managing a Resource Architecture Assembly Diagram 112

Creating a Resource Architecture Assembly Diagram: 112
Adding a Resource Architecture 112
Adding an Org-Unit or a Position Type 113
Describing Resource Architecture Technical Resources 113
Describing the Services in a Resource Architecture Assembly Diagram 114

Describing a Resource Architecture Environment 115

Creating a resource architecture environment 115
To create a resource architecture environment diagram 115
Describing IT Components 117

Describing an IT infrastructure 117

Creating an IT infrastructure 117
Creating an Infrastructure Assembly Structure Diagram 117
Using an Infrastructure Assembly Structure Diagram 117

Describing a Computing Device 118

Accessing the list of computing devices 118
Creating a computing device 119
Creating a Computing Device Assembly Diagram 119

Describing an IT network 120

Creating an IT network 120
Creating an IT network 120
Describing communications in a Technical Infrastructure. 122

Describing services and requests 122

Service points 122
Request points 122

Describing Communications at Equipment Level. 123

Communication ports 123
Communication channels 123
Network communication protocols 124

Using service catalogs 127
Introducing service catalogs 128

The types of service catalogs 128

Performing a functionality 129

Populating a service catalog 131
Creating an information service catalog 131
Accessing the list of service catalogs 132
Adding a service catalog item 132
Specifying the implementation of a service catalog item 132

| | |
|--------------------------------------|------------|
| Service catalog reports | 134 |
|--------------------------------------|------------|

Managing IT transformation 137

| | |
|---|------------|
| Overview of HOPEX IT Strategy | 138 |
| HOPEX IT Strategy Profiles. | 138 |
| <i>IT strategist.</i> | <i>138</i> |
| <i>Strategic Planning Functional Administrator.</i> | <i>138</i> |
| Connecting to the solution | 138 |
| HOPEX IT Strategy Desktop | 139 |
| <i>Presenting the IT Strategist desktop.</i> | <i>139</i> |
| <i>Presenting the functional administrator desktop.</i> | <i>140</i> |
| The HOPEX IT Strategy method | 140 |
| Starting with HOPEX IT Strategy. | 140 |
| Managing IT Architecture Transformation With HOPEX IT Strategy | 141 |
| Identifying and Assessing Motivations | 142 |
| Building and Adjusting the Transformation Roadmap. | 143 |
| <i>Defining enterprise stages.</i> | <i>144</i> |
| <i>Defining the enterprise and its events.</i> | <i>144</i> |
| Identifying Strategic Transformation Elements | 145 |
| Assessing Capabilities With HOPEX IT Strategy | 147 |
| Describing the Existing Architecture of Business Capabilities | 147 |
| <i>Describing a business capability.</i> | <i>148</i> |
| Defining the Functionalities Associated with Business Capabilities. | 148 |
| Assessing a Business Capabilities Map | 149 |
| Describing the target architecture with HOPEX IT Strategy | 150 |
| Describing a Logical Application System with HOPEX IT Strategy | 150 |
| <i>Accessing the list of logical application systems with HOPEX IT Strategy.</i> | <i>150</i> |
| <i>Describing a Logical Application System with HOPEX IT Strategy</i> | <i>151</i> |
| Describing Logical Applications with HOPEX IT Strategy. | 152 |
| <i>Accessing the list of logical applications with HOPEX IT Strategy</i> | <i>152</i> |
| Describing the Logical Application System Environment with HOPEX IT Strategy | 153 |
| Describing Implementation of a Business Capabilities Map. | 154 |

Using IT architecture diagrams 155

| | |
|---|------------|
| Creating a structure diagram | 156 |
| Creating an Application Structure Diagram | 156 |
| <i>The components of an Application Structure Diagram</i> | <i>156</i> |
| <i>Adding an application service to an application structure diagram</i> | <i>157</i> |
| Describing a scenario of flows | 158 |
| Using a Scenario of Application Flows Diagram | 158 |
| <i>Creating a Scenario of Application Flows diagram.</i> | <i>159</i> |
| <i>Adding an IT service to the scenario of application flows.</i> | <i>159</i> |
| <i>Managing application flows in a scenario of application flows</i> | <i>159</i> |
| <i>Adding an application data store to the scenario of application system flows</i> | <i>160</i> |

| | |
|--|------------|
| <i>Creating an application data channel</i> | 161 |
| Using a Scenario Sequence Diagram. | 161 |
| <i>Creating an application environment scenario sequence diagram</i> | 162 |
| <i>Instances of applications, IT services or interfaces</i> | 162 |
| <i>Message instance</i> | 163 |
| Describing a Technical Architecture | 164 |
| Creating an Application Technical Architecture Diagram | 164 |
| <i>Adding an application technical area to an application technical architecture diagram</i> | 165 |
| <i>Defining the software technologies used by an application technical area</i> | 165 |
| <i>Creating a technical communication line</i> | 166 |
| <hr/> | |
| Describing data exchanges | 167 |
| Managing Interactions | 168 |
| Creating an Interaction | 169 |
| Describing Service and Request Points | 169 |
| <i>Service points</i> | 169 |
| <i>Request points</i> | 170 |
| <i>Creating a Service Point or a Request Point</i> | 171 |
| Defining the Element Interaction Point | 171 |
| <i>Characterizing the element interaction point</i> | 171 |
| Describing Exchanges | 173 |
| Creating an Exchange. | 173 |
| Describing Exchanges. | 174 |
| <i>Creating an exchange diagram (BPMN)</i> | 174 |
| <i>Creating a message flow with content</i> | 174 |
| <i>Managing events, gateways and sequence flows</i> | 175 |
| Describing Exchange Contracts | 176 |
| Examples of Exchange Contract Diagrams (BPMN) | 177 |
| <i>Exchange contract diagram (BPMN)</i> | 177 |
| <i>Advanced communication exchange contract example</i> | 177 |
| Creating an Exchange Contract from an Interaction | 178 |
| Creating an Exchange Contract Diagram (BPMN) | 179 |
| Defining an Exchange or an Exchange Contract Use | 179 |
| <hr/> | |
| HOPEX IT Architecture Reports | 181 |
| Business Capabilities reports | 182 |
| Breakdown map of business capabilities | 182 |
| Matrix of Business Capabilities and Expected Functionalities | 184 |
| Reports on the Architecture Functional Coverage | 186 |
| Building Block Breakdown report | 186 |
| Matrix of Business Capabilities and Expected Functionalities | 188 |
| Application Architecture Reports | 190 |
| Application Architecture Breakdown Report | 190 |
| Impact report (Scenario). | 192 |

Impact report (Structure) 194

Infrastructure Reports196

Infrastructure Description Report 196

INTRODUCTION



HOPEX IT Architecture allows IT managers to formalize business needs in order to define the architecture of the information system that meets them, from the logical architecture to the infrastructure.

HOPEX IT Architecture offers different analysis perspectives:

- ✓ **Information System management and upgrading:** a description of service and city planning architectures are two approaches that simplify IS upgrading by providing a frame of reference for planning your systems and analyzing your upgrading scenarios (with **HOPEX Planning**). **HOPEX IT Architecture** proposes functions dedicated to representation and documentation of IT architectures using a service oriented approach.
- ✓ **Application mapping:** a description of application architecture that offers a detailed view of information exchanges between applications, services, databases and organizational units.
- ✓ **Application deployment:** a description of the information system infrastructure to monitor application deployment on the different enterprise sites. The infrastructure takes into account the main IT hardware of your organization such as networks, servers, workstations, printers, firewalls and concentrators.
- ✓ **The representation of resource architectures:** a description of complex systems involving different types of IT resources.

In addition to **HOPEX IT Architecture**, **HOPEX IT Strategy** allows organizations to manage their information system transformation programs. Each step of the transformation can be characterized by the IT resources implemented to cover the expected business capabilities, or functionalities.

The purpose of this guide is to present how to make best use of **HOPEX IT Architecture** for the successful evolution of your information system.

The following points are covered in **HOPEX IT Architecture**:

- ✓ "Modeling Business Capabilities and the logical application architecture", page 61.
- ✓ "Modeling Technical and Functional Architectures", page 81;
- ✓ "Modeling IT Infrastructures", page 111;
- ✓ "Managing IT transformation", page 137.
- ✓ "Using service catalogs", page 127.
- ✓ "Describing data exchanges", page 167.

PRESENTATION OF HOPEX IT ARCHITECTURE

Combined with the products of the **HOPEX** suite, **HOPEX IT Architecture** supports a methodology and the tools used to describe, analyze and plan your information system transformation.

The Scope Covered by HOPEX IT Architecture

The modules offered in standard mode are used to follow a top-down approach, beginning with a review of the business capabilities of the enterprise and its strategy, and ending with a precise definition of the components of the existing or future information system.

Each module addresses specific user profiles. Standard reports are offered to simplify analysis of the subjects handled.

The method described in this guide is represented by the modules described below.

☛ *The order of use of these modules is given by way of information.*

Describing the upgrade strategy of the information system: this step consists in describing what the information system is able to deliver, through business capabilities, and how it plans to deliver them using the architectures.

☛ *For more details, see ["Building the Business Capability Maps of the Organization"](#), page 15 and ["Building the Logical Architecture"](#), page 18.*

Describing the application system: this step consists in describing the application architecture and all the elements that compose it.

☛ *For more details, see ["Describing Application Architecture"](#), page 22.*

Describing and analyzing flows: this step is based on scenario diagrams that represent the flows between the components of your information system.

☛ *For more details, see ["Describing flow scenarios"](#), page 26.*

Analyzing the functional coverage of the application architecture: during this step the reports proposed by **HOPEX IT Architecture** are used to analyze the links between the components of the described application architecture and the expected functionalities.

☛ *For more details on modeling applications and services, see ["Describing Application Architecture"](#), page 22 and ["Defining the Application Technical Architecture"](#), page 27.*

Describing the infrastructure : this module allows to manage deployment constraints and to associate adapted solutions to them.

☛ *For more details, see ["Defining the infrastructure"](#), page 28.*

Describing the technical service catalogs: this last step consists in grouping in a service catalog the list of functionalities covered by the systems described.

☛ *For more details, see ["Managing Service Catalogs"](#), page 34.*

HOPEX IT Strategy, which is an option of **HOPEX IT Architecture**, is based on the tools of the **HOPEX** platform as well as on the method embedded in the **HOPEX**

Business Architecture solution to support the description, analysis and resources of information system transformation projects.

➡ For more details, see *"Managing IT transformation"*, page 137.

Summary of Activities and Deliverables of HOPEX IT Architecture

Activities are associated with each of the modules of the method we recommend for managing the evolution of your information system.

The **HOPEX IT Architecture** solution offers the tools to carry out these activities, which are materialized by deliverables.

| Activities | Main deliverables |
|---|--|
| Defining the logical architecture | Logical architecture structure diagrams, see "Describing a Logical Application Architecture with HOPEX IT Architecture" , page 71. |
| Building the application architecture | Application architecture structure diagrams and flow scenario diagrams, see "Describing Application Architecture" , page 82 and "Describing Application Flows" , page 100. |
| Analyzing the functional coverage of the application architecture | Assessing the functional coverage by software resources, see "Analyzing the functional coverage of the application architecture" , page 32. Assessment of the coverage of technical functionalities by technical resources, see "Analyzing the Functional Coverage of the Application and Hardware Architecture" , page 33. |
| Defining the technical architecture | Description of the technical requirements for the application deployment, see "Describing Technical Architecture" , page 105. |
| Defining the infrastructure | Description of the technical requirements for the application deployment, see "Modeling IT Infrastructures" , page 111. |
| Managing service catalogs | Description of service catalogs and recommended solutions, see "Using service catalogs" , page 127. |

Presentation of the HOPEX IT Architecture deliverables

Structure and positioning of the HOPEX IT Architecture Solution

HOPEX IT Architecture can be used with other products in the **HOPEX** suite.

HOPEX Business Process Analysis

The **HOPEX Business Process Analysis** solution provides **HOPEX IT Architecture** with the possibility to describe the organizations and processes that implement the business capabilities identified in **HOPEX IT Architecture**;

HOPEX Risk Mapper

The **HOPEX Risk Mapper** solution provides **HOPEX IT Architecture** with the possibility to associate risks with elements of the information system.

HOPEX Project Portfolio Management

The **HOPEX Project Portfolio Management** solution provides **HOPEX IT Architecture** with the possibility to analyze the potential return of a set of projects to monitor the alignment of projects with the organization's strategic objectives and ensure consistency between projects and the organization's capacity.

HOPEX Information Architecture

The **HOPEX Information Architecture** provides **HOPEX IT Architecture** with the possibility to build the global architecture, from business data definition to database design. This solution ensures the traceability of data between three different levels: conceptual, logical and physical.

HOPEX IT Architecture Profiles

In **HOPEX IT Architecture**, there are, by default, business profiles with which specific activities are associated.

Presentation of the solution interface depends on the profile selected by the user on connection to the application; the tree of menus and functions varies from one business role to another.

☛ For more details on the Desktops connected to each of the profiles, see ["HOPEX IT Architecture Desktop Presentation", page 36.](#)

| Profiles | Tasks |
|---|---|
| IT Architecture Functional Administrator | In addition to the functional rights of the application manager, the IT Architecture Functional Administrator has rights to all objects, methods, projects and workflows. He/she prepares the work environment and creates the elements required to manage the modeled elements. Manages: - Users, assignment of roles and access rights to the different project steps. - all the environment objects (application environments, infrastructures, reports, etc.), - Workflows. For more details, see "Presentation of the IT Architecture Functional Administrator space", page 38. |
| IT Architecture Manager | The IT Architecture Manager has rights to all objects, methods, projects and assessments. For more details, see "Presenting the IT Architecture Manager space", page 41. |
| Application architect | The application architect is responsible for building the application architecture models assigned to him/her. He/she can manage transformation projects. For more details, see "Presentation of the Application Architect space", page 41. |
| IT infrastructure architect | The IT infrastructure architect is responsible for building the infrastructure and hardware models assigned to him/her. He/she can manage transformation projects. For more details, see "Presentation of the Infrastructure Architect space", page 42. |
| IT Technical Architect | The IT Technical Architect is responsible for building the technical architecture models assigned to him/her. He/she can manage transformation projects. For more details, see "Presentation of the IT Technical Architect space", page 43. |
| Applications Contributor and Application Contributor (lite) | The applications contributor is responsible for validating the design of the applications assigned to him/her. For more details, see "Presentation of the Application Contributor space", page 44. |
| Applications Viewer and Applications Viewer (lite) | Applications viewer and Applications Viewer (lite) have read-only rights on objects in the repository. For more details, see "Presentation of the Application Viewer space", page 44. |

Business Roles of HOPEX IT Architecture

In **HOPEX IT Architecture**, there are, by default, business roles that can be assigned to certain users. These roles are:

- **Software Designer**: used to assign a user to software elements. The software designer is responsible for designing the software assigned to him/her.
- **Local Application Owner** used to assign a user to applications. The Local Application Owner is responsible for the following tasks:
 - Identifies risks
 - Responding to Questionnaires
 - Defining and implementing action plans,
 - Validating the modifications made by the architect in the context of object review workflows.
- **IT Owner**: specifies the characteristics of IT elements for which he/she is responsible.
- **Business Owner**: specifies the characteristics of software and applications for which he/she is responsible.

THE HOPEX IT ARCHITECTURE METHOD


Building the Business Capability Maps of the Organization

The goal of this step, on a strategic level, is to check the suitability between the business capabilities of the enterprise, the functionalities covered and the applications that deliver them.


For more details on managing business capabilities with **HOPEX IT Architecture**, see ["Using Business Capabilities with HOPEX IT Architecture"](#), page 62.

Describing Business Capabilities

A **business capability** defines an expected skill.

 A **business capability** is a set of features that can be made available by a system (an enterprise or an automated system).

A **business capability map** describes what the enterprise is capable of producing for its internal needs or for meeting the needs of its clients. It is thus based on the main business capabilities of its activity at a given moment.

 A **business capability map** is a set of business capabilities with their dependencies that, together, define a framework for an enterprise stage.

For example, the standard ability to manage "Operational Activities" is based on the business capabilities to process "Supply", "Sales" and "Complaints", "Order Management" and "Customer Management".




Example of a business capability map


For more details on managing a business capability map, see ["Using Business Capabilities with HOPEX IT Architecture"](#), page 62.

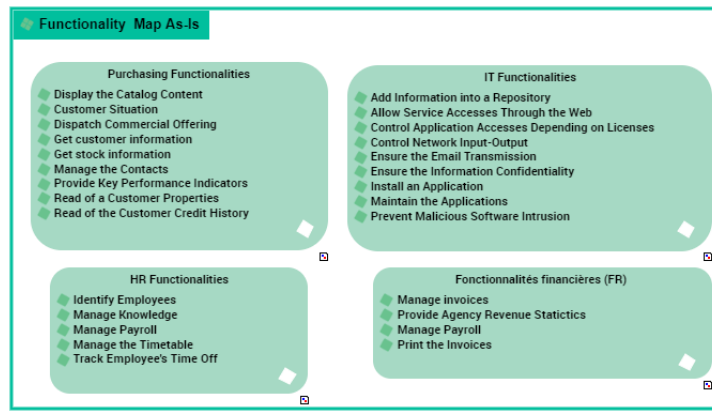
Identifying the functionalities associated with business capabilities

A **functionality** defines an ability necessary for the proper functioning of the IS to cover business expectations.

 A **functionality** is a service required by an org-unit in order to perform its work. This functionality is generally necessary within an activity in order to execute a specific operation. If it is a software functionality, it can be provided by an application.

A **functionality map** describes all the functionalities the enterprise is able to cover for its internal needs or for meeting the needs of its clients.

 A **functionality map** is a set of functionalities with their dependencies that, jointly, define the scope of a hardware or software architecture.

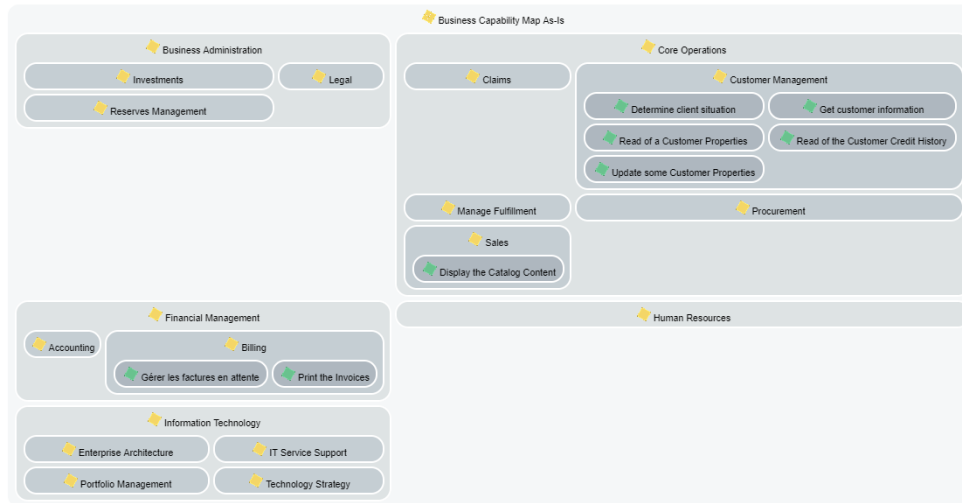


Example of a functionality map

 For more details on managing a functionality map, see ["Describing functionalities with HOPEX IT Architecture", page 68.](#)

The description of *business capabilities* and *functionalities* is particularly interesting if business capabilities are associated with the functionalities that fulfill them.

1. Business Capability Map Breakdown



Example of a business capability breakdown report

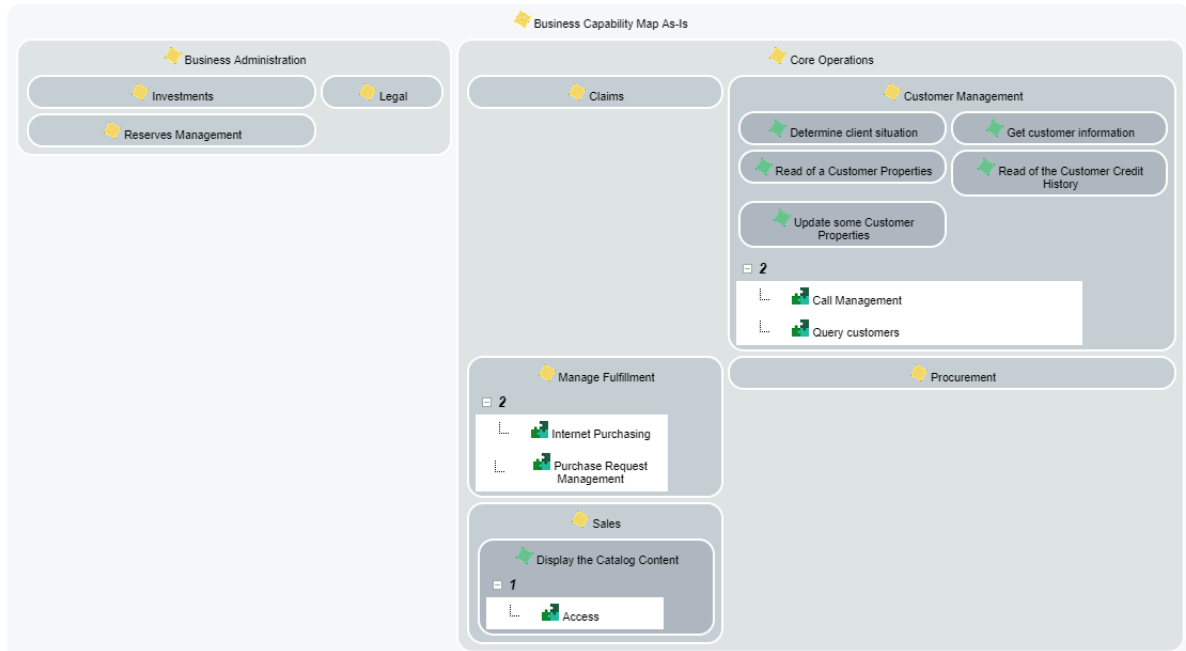
For more details on this breakdown report, see ["Breakdown map of business capabilities"](#), page 182.

Identifying the applications associated with business capabilities

Applications cover *functionalities* associated with *business capabilities*. In **HOPEX IT Strategy**, a report allows to check the functional coverage of your applications.

🔖 For more details on this breakdown report, see ["Creating a business capability realization"](#), page 65.

1. Business Capability Map Breakdown



Building the Logical Architecture


HOPEX IT Architecture provides ways to define logical application architectures that represent ideal architectures. These representations make it possible to design logical structures for application architectures, to rationalize exchanges between these structures and to identify the data used. Logical application architectures can then be compared with the implemented architectures to detect gaps between the real and the ideal.

🔖 For more details on use of a logical application system, see ["Describing a Logical Application Architecture with HOPEX IT Architecture"](#), page 71.

Logical application systems can be described using a top-down approach, starting with a description of the company's main application systems, or a unitary approach, describing only some logical application systems.


📖 A logical application system is an assembly of other application architectures, logical applications and end users, interacting with application components to implement one or several functions.


If you use a unitary approach, you will need to describe the *logical application system environment* to describe the context in which the logical system is used and its interactions with external logical components.

 A logical application system environment presents a logical application system use context. It describes the interactions between the logical application system and its external partners, which allows it to fulfill its mission and ensure the expected functionalities.

At this level of the method, this step, which is not mandatory, covers the following points:

- Identify the exchanges between the different logical components and formalize them through *exchange contracts*.

 An exchange contract is a model of a contract between organizational entities. This contract is described by exchanges between an initiator role and one or several contributor roles.

 For more details on exchange contracts, see "[Describing data exchanges](#)", page 167.

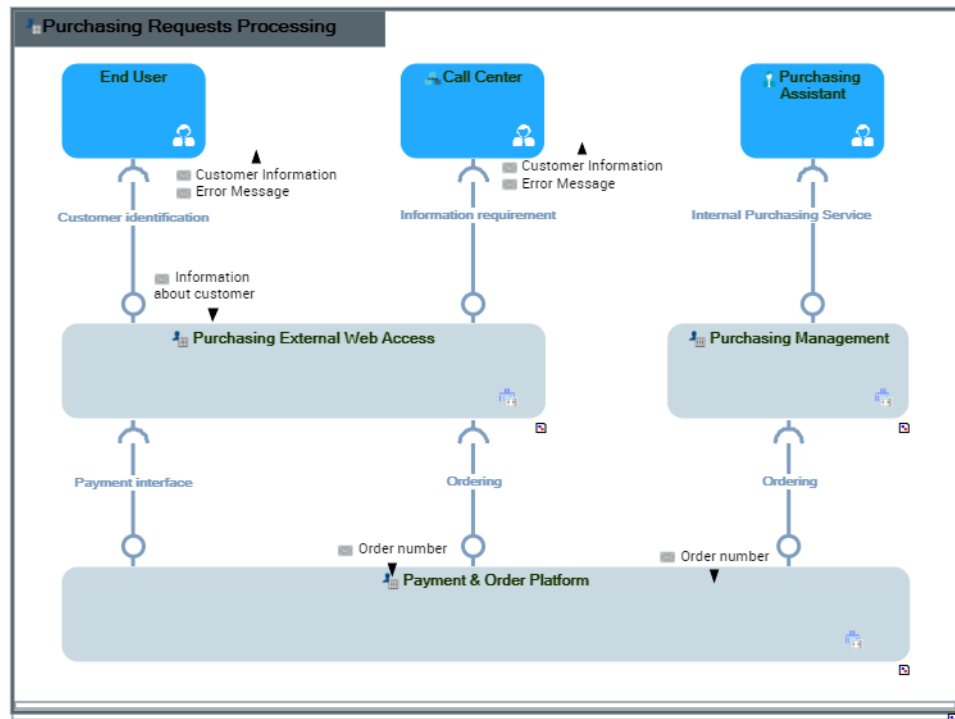
- Verify that the logical architecture covers the functional requirements identified in the business capability maps.

Structure diagram of the logical application system

The components of the *logical application system* are described in a structure diagram of the logical application system that presents:

- the services offered or required;
- the processes handled, components and their interactions;
- the end users interacting with the application components.

The following diagram describes the structure of the logical application system "Purchase request processing" offered to customers.



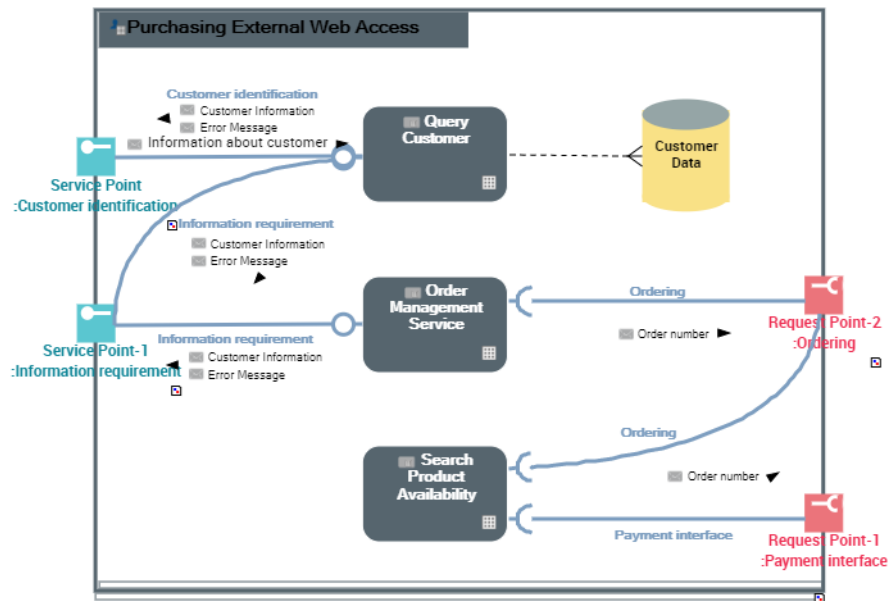
Structure diagram of the "Internet Purchasing" logical application system

"Internet Purchase Requests" are offered to customers either directly or through a "Call Center".

Requests made by customers are processed by a "Internet Purchase Requests" logical application system.

The logical application system structure diagram, for managing "Internet Purchase Requests", presents different

logical applications, access to a logical database as well as service and request points for "Book" or "Order".



Structure diagram of the "Management Shopping Cart" logical application system

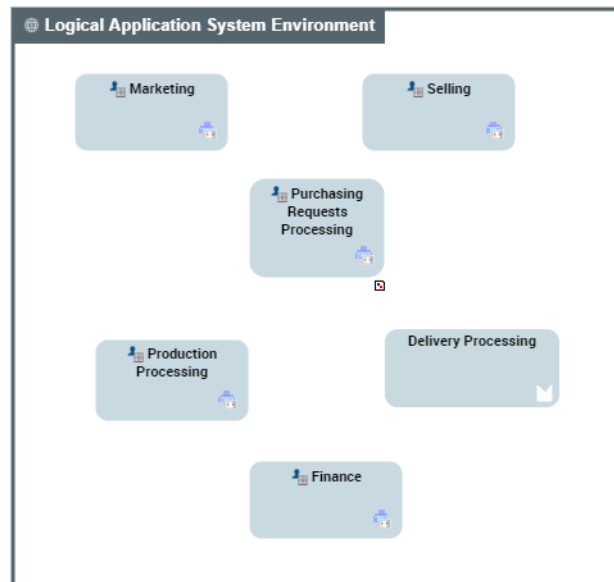
For more details on use of a logical application system, see ["Describing a Logical Application System with HOPEX IT Architecture", page 71.](#)

Logical application system environment diagram

A logical application system environment presents a logical application system use context. It describes the interactions between the logical application system and its external partners, which allows it to fulfill its mission and ensure the expected functionalities.

The components of a *logical application system environment* are presented in an application system environment diagram that describes the internal logical application systems as well as the partner logical application systems.

s



Logical application system environment diagram

The internal logical application system "Purchase request processing" uses a logical "Delivery" application system that is external to the described environment.

➤ For more details on use of a logical application environment system, see ["Logical Application System Environment Description", page 76.](#)

Describing Application Architecture

HOPEX IT Architecture offers the means to represent different levels of application architectures: from the description of the application environment to the technical components to be implemented.


These representations make it possible to define the software and hardware components and to identify in a consistent way the data exchanged between them.

➤ For more details on the use of an application architecture, see ["Modeling Technical and Functional Architectures", page 81.](#)

The description of *application systems* can be done according to a top-down approach, starting by describing the company's main application systems, or according to a unitary approach by describing only certain application systems.


📖 An application system is an assembly of other application systems, applications and end users interacting with application components to implement one or several functions.


If you use a unitary approach, you will need to describe the *application system environment* to describe the context in which the application system is used and its interactions with external components.

 *An application system environment allows presenting the other application systems, applications or micro-services with which this application system can interact.*

In addition to a precise description of the application architecture to be implemented, this step covers the following points:

- Identify precisely the exchanges between the different software and hardware components and formalize them through *exchange contracts*.

 *An exchange contract is a model of a contract between organizational entities. This contract is described by exchanges between an initiator role and one or several contributor roles.*


 *For more details on exchange contracts, see ["Describing data exchanges"](#), page 167.*

- Verify that the application architecture covers the functional requirements identified in the business capability maps.


 *For more details on the functional analysis, see ["Analyzing the Functional Coverage of the Architecture Implemented"](#), page 32.*

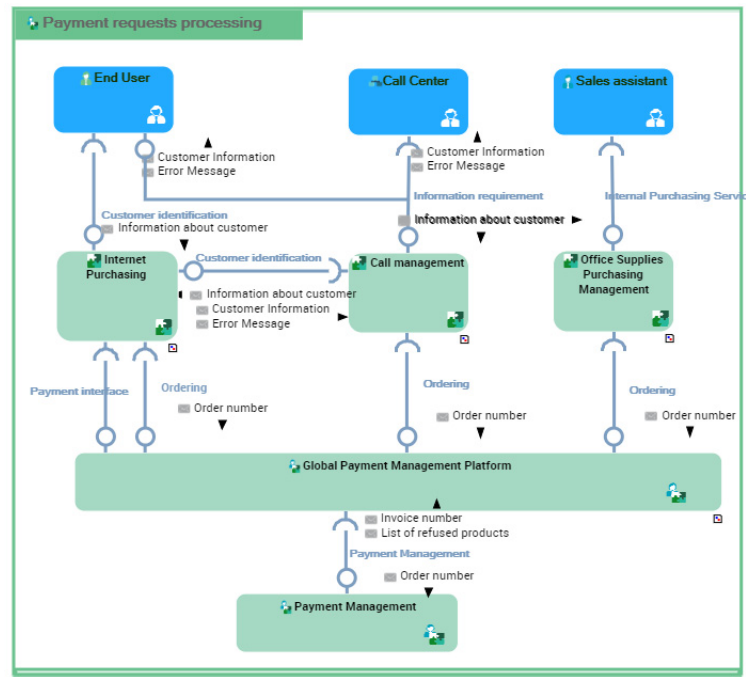
Describing application systems

In an approach where application systems are studied in a unitary manner, it is preferable to create application environments to describe the context in which the systems described are used.

 *For more details on application system environments, see ["Describing an Application System Environment with HOPEX IT Architecture"](#), page 90.*

In a top-down approach, the main application system structure diagram is the entry point for the description of the existing or planned application system.

 *For more details on application systems, see ["Describing an Application System"](#), page 86.*



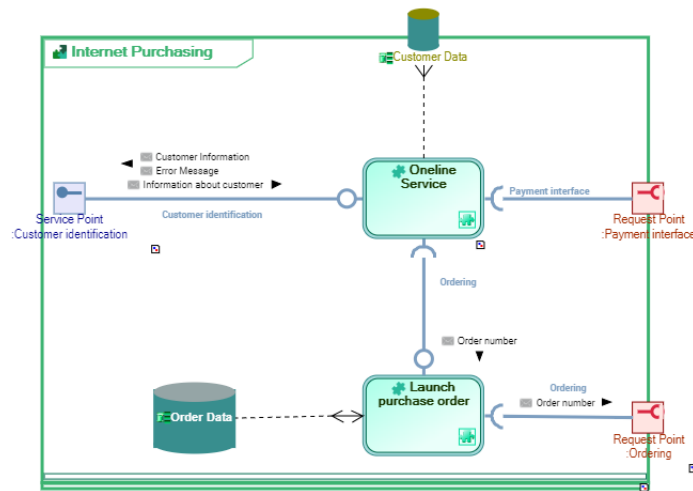
The following diagram describes the application system corresponding to purchasing request processing.

Purchase requests are formulated by private users or by companies in different conditions.

The "Purchasing Request Processing" application system offers an Internet purchasing service.

The application subsystems can then be described hierarchically by showing at each level the points of exchange with the outside world.

The "eCommerce purchase" application system, for order management, manages order data in an internal data store and customer data in an external data store.



Application system structure diagram of the "Internet Purchasing" sub-system

The **data stores** are used to represent the data that will be stored in databases.

A data store provides a mechanism to update or consult data that will persist beyond the scope of the current process. It enables storage of input message flows, and their retransmission via one or several output message flows.

For more information on data stores, see ["Managing Data", page 96](#).

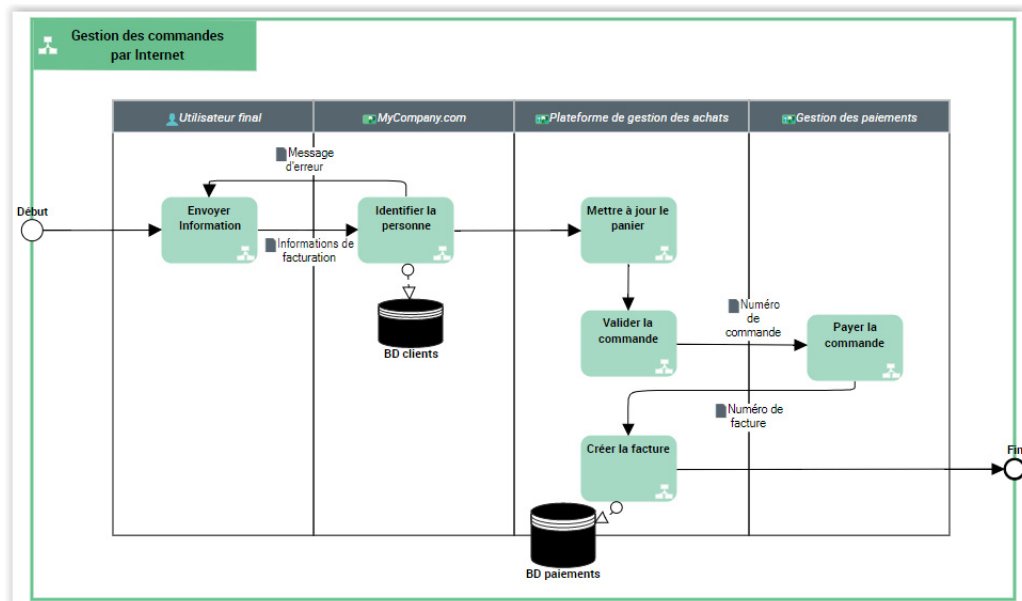
Describing application processes

HOPEX IT Architecture offers the possibility to check that the processes performed by the application system are correctly covered by describing **Application processes**.

A system process is the executable representation of a process. the events of the workflow, the tasks to be carried out during the processing, the algorithmic elements used to specify the way in which

the tasks follow each other, the information flows exchanged with the participants.

For more details on system processes, see ["Describing Application Processes"](#), page 94.



Application process diagram

Describing flow scenarios

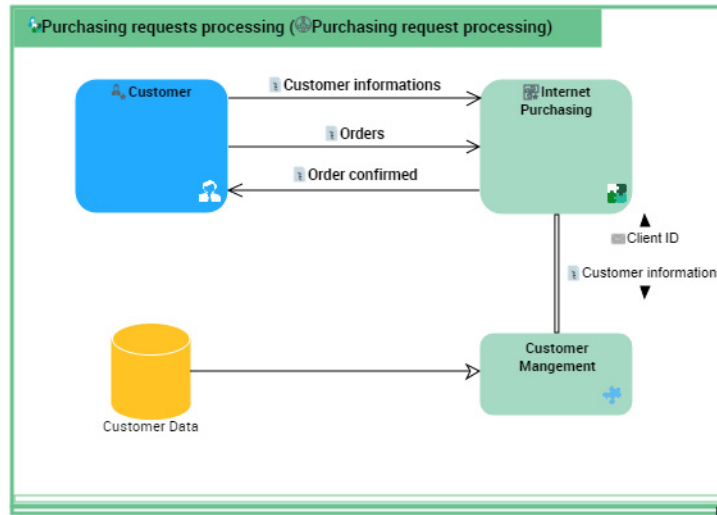
Structure diagrams use interactions to describe the data exchange capabilities between components. In addition, the flow scenarios make it possible to precisely describe the main use cases and implementation of these data exchanges.

It is possible to define flow scenarios at all levels of the application architecture:

- Application
- Application System
- Application Environment
- IT service
- Micro-service

The objective of flow scenarios is to verify that content is correctly conveyed between components.

The scenario of application system flows below describes the interactions between a client and the eCommerce application.



Example of scenario of application system flows for "Purchasing Requests Processing".

For more details on flow scenarios, see ["Describing Application Flows"](#), page 100.

Defining the Application Technical Architecture

An application technical architecture allows to represent packages to be deployed in the form of *application technical areas* and *technical data areas* as well as the *technical communication lines* necessary for connections between areas (protocols).

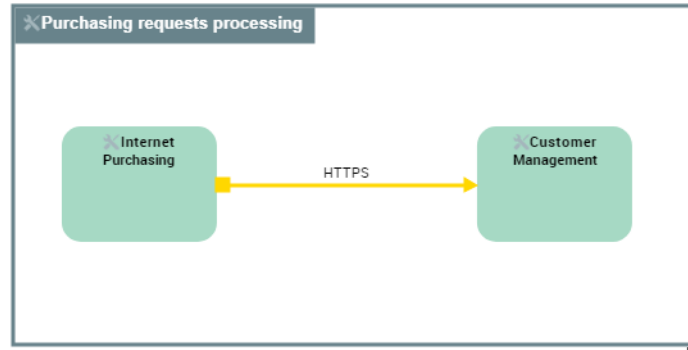
An application technical area represents an organizational element of an application according to technical criteria. For example, this can be the user interface or a process. Each application technical area is associated with one or more technologies. The deployment of several application technical areas is necessary for the application to be operation.

A data technical area represents an organizational element of an application used to access the data necessary for the operation of this application. Each application technical area is associated with one or more technologies (E.g.: Oracle 12, SQL Server 2012, etc.). A data technical area can allow access to one or more data stores.

A technical communication line represents a technical connection between architectures or application technical areas through client and server ports. Client technical port of an architecture or a technical area

requires opening the communication line to server technical port of the other area or technical architecture.

The application technical architecture below presents the application technical areas for eCommerce and customer management. The technical communication line between these two components is based on the https communication protocol.



Example of the application technical architecture for "Purchasing Request Processing".


For more details, see ["Describing Technical Architecture", page 105.](#)

Defining the infrastructure

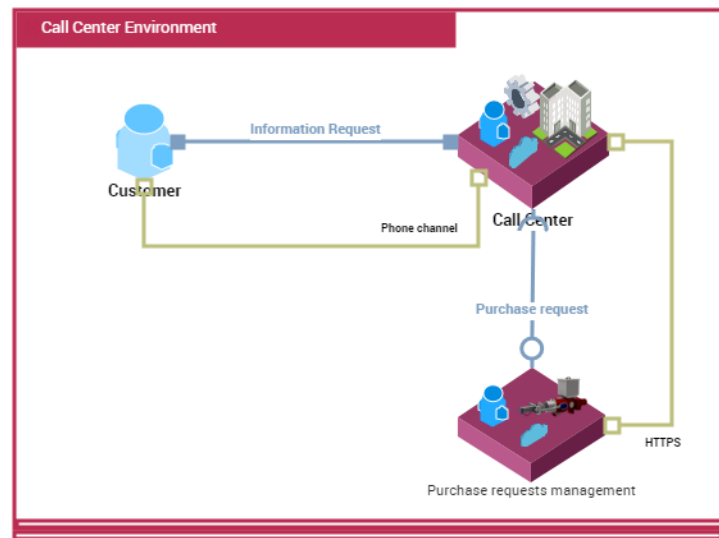
The infrastructure is made up of all the elements required to host and operate the application components and technologies.

With **HOPEX IT Architecture** the infrastructure can be described in a bottom-up approach, from the most detailed to the most conceptual, or top-down, from the most conceptual to the most detailed. Presentation of these functionalities is based on the example of a call center.

Resource Architecture Environment Diagram

 A business architecture environment represents the relationships of a business functional area with its partners.

The following diagram describes the environment of a support center.



The call center responds to customer requests. It is based on an external service to fulfill any purchasing requests.

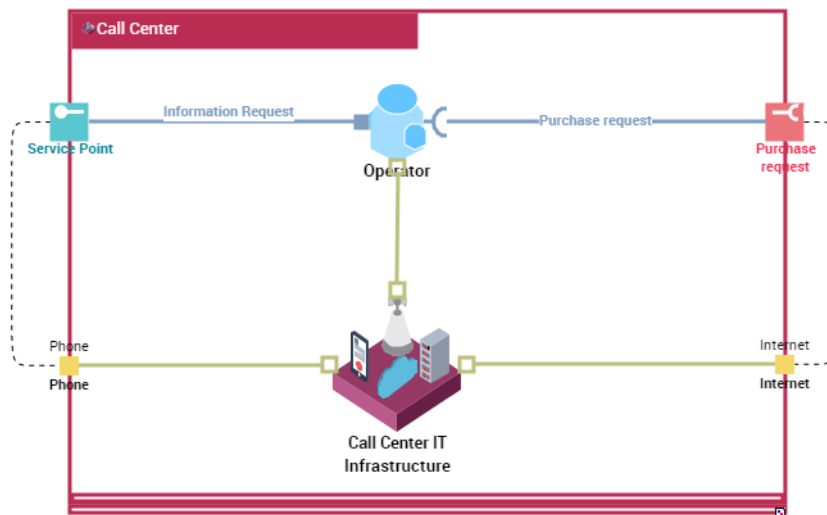
☛ For more details, see ["Describing a Resource Architecture Environment", page 115.](#)

Describing Resource Architectures

The Resource Architecture Assembly Diagram describes the hardware and organizational resources required for handling service requests.

☛ For more details, see ["Describing a Resource Architecture", page 112.](#)

In the call center example, we consider only the operator and the IT infrastructure that represents its equipment.



A team of operators handles all requests, whatever their nature, by telephone or by e-mail.

The operator identifies the caller, records the request, applies a first filter (in case of error) and if necessary records a purchasing request via request points.

This diagram contains a *Request Point* from which the operators make purchasing requests.



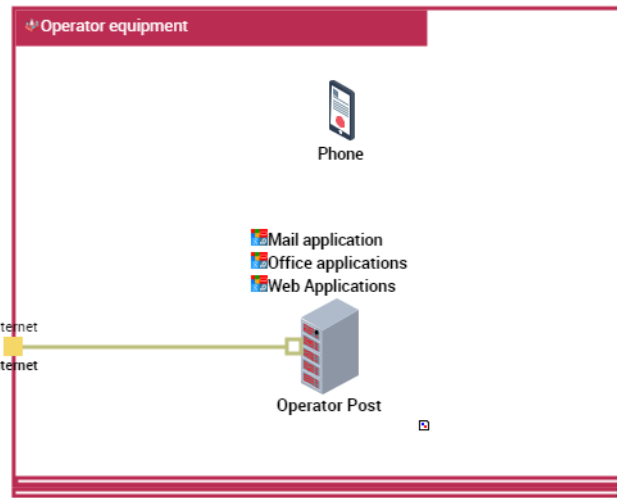
A request point is a point of exchange by which an agent requests a service from potential suppliers.

IT infrastructure assembly structure diagram

This diagram presents an IT infrastructure. It contains Infrastructure IT component such as: computers or IT equipments.




For more details, see ["Describing an IT infrastructure", page 117.](#)



The basic hardware architecture of a call center includes two link points to the outside: a telephone link, a link to a private network that enables the HTTP link for the purchasing request.

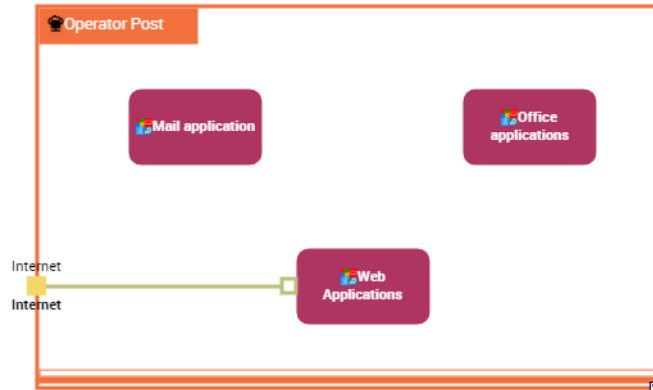
Note that the *Communication Protocols* used are specified on the communication channels.

 A communication protocol is a set of standardized rules for transmission of information (voice, data, images) on a communication channel. The different layers of protocols can handle the detection and processing of errors, authentication of correspondents, management of routing.

Computing Device Assembly Diagram

The computing device assembly diagram presented below describes the software technologies, the application technical area and IT devices installed on a standard PC.

☛ For more details, see ["Describing a Computing Device", page 118](#).



Computing device assembly diagram of a standard PC.

A standard PC is equipped with office system applications and electronic mail applications.

A standard PC also has an HTTP connection to access Web applications to manage purchase requests.

Analyzing the Functional Coverage of the Architecture Implemented

The goal of this step is to check the adequacy between the expected functionalities and the architecture elements that deliver them.

Analyzing the functional coverage of the application architecture

The aim here is to connect the *functionalities*, which correspond to what is expected to achieve the objectives, to the means of implementation represented by *applications* or *application systems*.

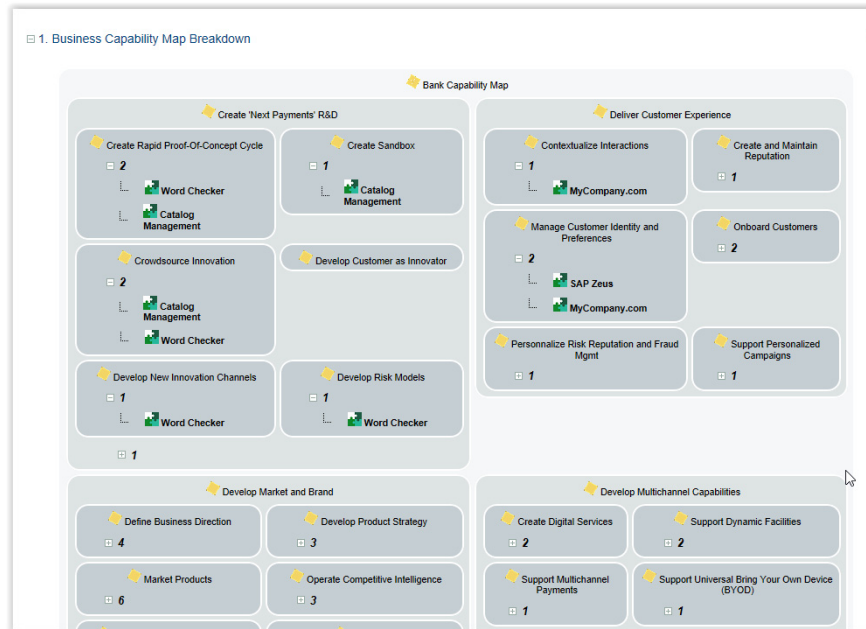
📖 An application system is an assembly of other application systems, applications and end users interacting with application components to implement one or several functions.

By constructing the *functionality map* on the one hand and the *application system environment* on the other hand, you can check that the functionalities are implemented by application components.

☛ For more details on how to associate a functionality with an application component, see ["Describing Implementation of a Functionality", page 69](#).

HOPEX IT Architecture provides a report that presents the result of the implementation of business capabilities by logical or physical applications (or application systems).

For more details on how to associate a business capability with an application, see *"Creating a business capability realization"*, page 65.



Example of a map for the implementation of capabilities by applications

Analyzing the Functional Coverage of the Application and Hardware Architecture

The aim is to connect the *functionalities*, which correspond to what is expected to achieve the objectives, to the technical means of implementation represented by : *servers* or *devices*.

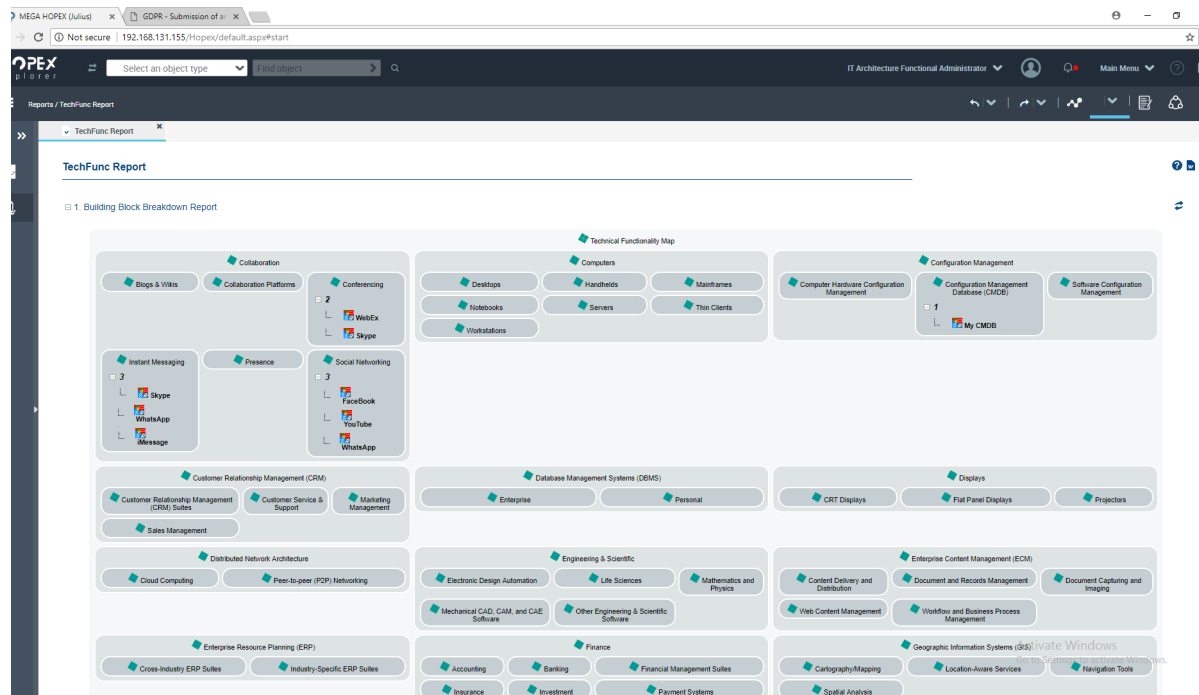
An IT Server is an IT component providing a service to users connected via an IT network. This IT component can house databases and run applications.

By constructing the *Functionality map* on the one hand and the *application technical architecture* on the other hand, you can check that the functionalities are implemented by technical components.

An application technical architecture describes one of the configurations possible for application deployment. It describes how the different technical areas of the application are connected to each other and the technologies and the communication protocols that they use. An application can have a number of possible technical architectures (E.g.: autonomous installation, horizontal or vertical deployment, etc.)

For more details on technical elements, see *"Describing Technical Architecture"*, page 105.

HOPEX IT Architecture provides a report that presents the result of the implementation of functionalities by *computing devices*.



Example of a map of functionalities implemented by technologies

For more details on how to associate a functionality with technical component, see ["Describing Implementation of a Functionality", page 69](#).

Managing Service Catalogs

A service catalog describes the list of functionalities covered by a solution as well as the technical or functional elements that implement these functionalities.

A service catalog contains a list of key service offers for which solutions are recommended.

HOPEX IT Architecture offers the following service catalogs:

- *technical service catalogs,*

A business service catalog provides a centralized information source for the business services offered by the service provider organization. It contains a customer-oriented view of the services associated to business capabilities, how they are supposed to be used, the processes that they support as well as the expected service quality level. The business service catalog presents the list of functionalities mentioned as well as implementation recommendations.

- *information service catalogs,*

An information service catalog provides a centralized information source for the information services offered by the service provider

organization. It contains a customer-oriented view of the information services used, how they are supposed to be used, the processes that they support as well as the expected service quality level. The information service catalog presents the list of functionalities mentioned as well as implementation recommendations.

- **technical service catalogs,**



A technical service catalog provides a centralized information source for the technical services offered by the service provider organization. It contains a customer-oriented view of the technical services used, how they are supposed to be used, the processes that they support as well as the expected service quality level. The technical service catalog presents the list of IT functionalities mentioned as well as implementation recommendations.

- **hardware service catalogs.**



A hardware service catalog provides a centralized information source for the hardware services offered by the service provider organization. It contains a customer-oriented view of the hardware used, how they are supposed to be used, the processes that they support as well as the expected service quality level. The hardware service catalog presents the list of hardware functionalities mentioned as well as implementation recommendations.

A specific report enables viewing of the functionalities covered by several service catalogs as well as elements that implement these functionalities.

1. Services coverage matrices

Retail Corp - IS Service Catalog

| | Description | proposed solution 1 |
|---|--|--|
|  Client & Customer Billing | Billing upon reservation must be available via mobile device |  Billing |
|  Consult Prices list | |  Catalog Management |

Retail Corp - Technical Service Catalog

| | Description | proposed solution 1 |
|---|-------------|--|
|  Application Server Functionalities | |  Retail Corp Central Infrastructure |

Example of service catalog report

For more details on the use of service catalogs, see ["Using service catalogs", page 127](#).

HOPEX IT ARCHITECTURE DESKTOP PRESENTATION

☛ **HOPEX IT Architecture** is mainly intended for web users. Desktops described in this guide are accessible only to Web desktop users.

If you have a **HOPEX IT Strategy** module, a specific desktop is available to you.

☛ For more details on the profiles and the facilities offered by the **HOPEX IT Strategy** module, see ["Overview of HOPEX IT Strategy", page 138](#).

Connecting to the solution

To connect to **HOPEX IT Architecture**, see **HOPEX Common Features**, "HOPEX Web Front-End Desktop".

HOPEX IT Architecture Desktop Presentation

The menus and commands available in **HOPEX IT Architecture** depend on the product licenses that you have and on the profile with which you are connected.




☛ For more details on using the Web platform for HOPEX solutions, see the **HOPEX Common Features** guide.

Presentation of space common to all profiles

All users, with the exception of users connected with the **Applications Contributor** and **Applications Viewer** profiles, have a **HOPEX IT Architecture** desktop and access to the following panes:

- **Home**: presents the main tiles useful for the user.
- **Dashboards**: displays the list of indicators required to steer objects such as processes, applications or org-units.
- **My Work**: displays the tasks assigned to the user.
- **Ideation**: used to manage the project portfolios and access specific reports, if you have the product license.
 For more information on project portfolio management, see the "Managing projects" section in the **HOPEX Common Features** guide.
- **Reports**, enabling access to the group of reports available for each solution.
 For more details on the use of these reports, see "Generating Reports" chapter in **HOPEX Common Features** guide.
 For more information on **HOPEX IT Architecture** reports, see "HOPEX IT Architecture Reports", page 181.
- **Collaboration**, which enables access to all collaborative tools provided by **HOPEX**.
 For more details on the use of collaborative tools, see "Accessing collaboration in HOPEX" chapter in the **HOPEX Common Features** guide.

The tiles of the **Home** space of **HOPEX IT Architecture** are:

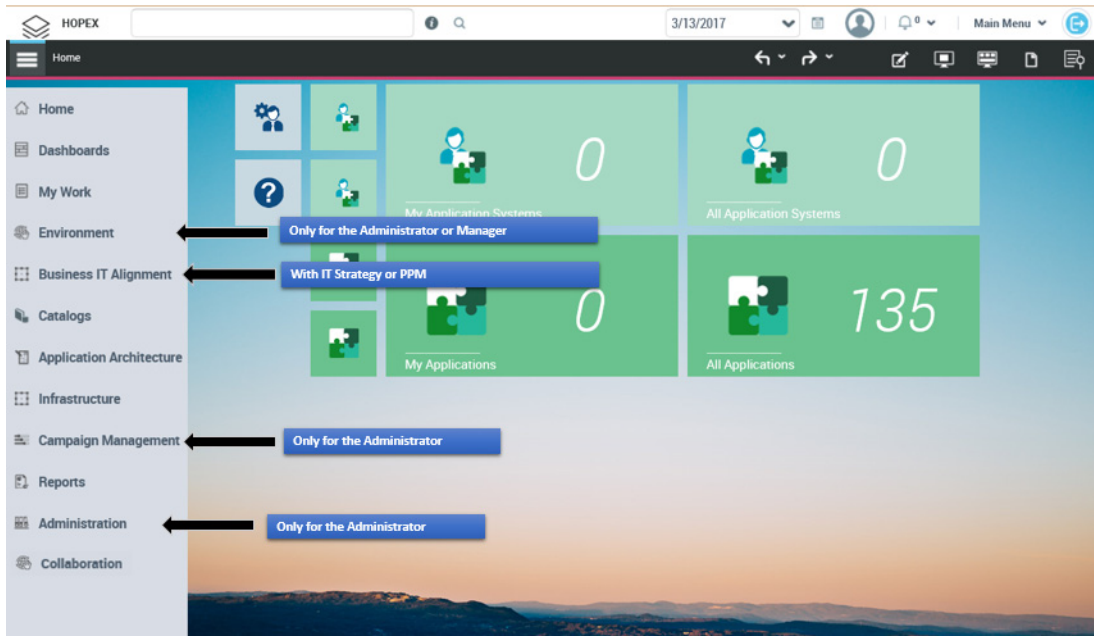
- **Help**  to access the **HOPEX** documentation.
 - **Option**  to access to **HOPEX** options.
 - A add tile button .
- See "Adding a tile to your home page" chapter in the **HOPEX Common Features** guide.

The panes provide access to the following menus:

- **History**, which contains all the objects you access or modify.
 For more details on the use of history, see "Using the History" chapter in **HOPEX Common Features** guide.
- **Favorites**, to access to important objects and to usual actions.
 For more details on the use of favorites, see "Managing Favorites" chapter in the **HOPEX Common Features** guide.

Presentation of the IT Architecture Functional Administrator space

In addition to the panes offered in standard mode to all **HOPEX IT Architecture** desktop users, users connected with the **IT Architecture Functional Administrator** profile have access to the panes described below:




The Environment pane

The **Environment** pane provides access to the following menus.

- **Organization**, to access the main objects processed with the **HOPEX IT Architecture** solution.
- **Standard Navigation**, to access the management features for libraries and Environments.
 - ☛ For more details, see *"Preparing the Work Environment"*, page 46.
- **EA Projects**, to access the management features for projects.
 - ☛ For more information on project portfolio management, see the *"Managing project portfolios"* section in the **HOPEX Common Features** guide.


The Business IT Alignment pane

The **Business IT Alignment** pane provides access to the following menus:

- **Logical Architecture**, to describe the Logical Architecture elements.
 For more details on logical architectures, see ["Modeling Business Capabilities and the logical application architecture"](#), page 61.
- **Portfolios**, to access the project portfolio management functionalities offered with the **HOPEX Portfolio & Planning** product.
 For more information, see **HOPEX Portfolio & Planning**.
- **IT Architecture Projects**, to manage architecture projects.
 For more details on managing projects, see the ["Projects in HOPEX"](#) chapter in the **HOPEX Common Features** guide.
- **Corrective Action Plans**, to describe and manage the action plans linked to the transformation of processes.
- **Controls & Risks**, for accessing the risk management features offered with the **HOPEX Risk Mapper** product.
 For more information, see **HOPEX Risk Mapper**.




The Catalogs pane

The **Catalogs** pane provides access to the following menus:

- **Business Service Catalogs** to describe the user services offered by the information system.
- **Information Service Catalogs** to describe the user services offered by the information system.
- **Technical Service Catalogs** to describe the services offered by the technical elements of the information system.
- **Hardware Service Catalogs** to describe the services offered by the hardware elements of the information system.
- **Reports** to provide access to all the reports dedicated to the service catalogs.
 For more details on service catalogs, see ["Using service catalogs"](#), page 127.



The Application Architecture pane

The **Application Architecture** pane provides access to the following menus:

- **Functional Architecture** to describe the functional elements of the information system.
 For more details on functional and technical architectures, see ["Modeling Technical and Functional Architectures", page 81.](#)
- **Technical Architecture** to describe the technical elements of the information system.
 For more details on technical elements, see ["Describing IT Components", page 117.](#)
- **Controls & Risks**, for accessing the risk management features offered with the **HOPEX Risk Mapper** product.
 For more information, see **HOPEX Risk Mapper**.
- **Data Management** to access the management functionalities for business data offered with the **HOPEX Information Architecture** product.

The infrastructure pane

The **Infrastructure** pane provides access to the following menus.


- **Infrastructure** to describe the elements that make up the resource architecture.
 For more details on infrastructures, see ["Modeling IT Infrastructures", page 111.](#)
- **Technical Architecture** to describe the applications and the technology stacks of the information system.
 For more details on functional and technical architectures, see ["Modeling Technical and Functional Architectures", page 81.](#)

The Campaign Management pane

The **Campaign Management** pane provides access to the creation and management functionalities.

The Collaboration pane

The **Collaboration** pane makes available different means of communication in your **HOPEX IT Architecture** desktop.

 For more details on collaboration tools, see the ["Accessing collaboration in HOPEX"](#) chapter in the **HOPEX Common Features** guide.

The Administration pane

The **Administration** pane provides access to the user management features. The rights of different users on objects of imported libraries depend on their assigned profiles.

 For more details on the management users, see ["Managing users"](#) chapter in [guide HOPEX Common Features](#).

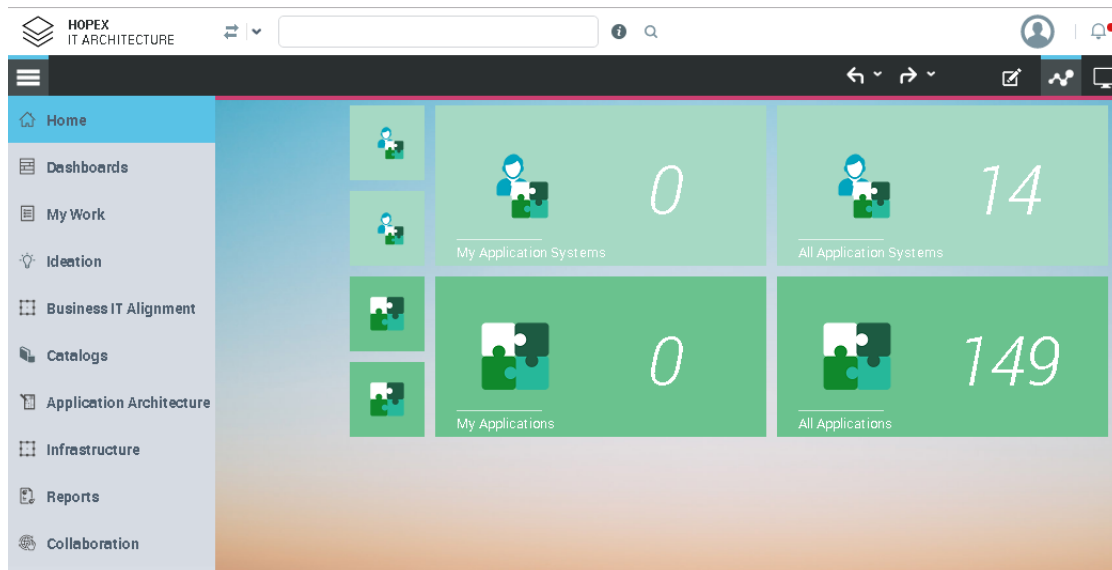
Presenting the IT Architecture Manager space

Users connected with the **IT Architecture Manager** profile have access to the same functionalities as users connected with the **IT Architecture Functional Administrator** profile. Only the administration activities are not offered.

For more details, see ["Presentation of the IT Architecture Functional Administrator space"](#), page 38.

Presentation of the Application Architect space

In addition to the panes offered in standard mode to all **HOPEX IT Architecture** desktop users, the **Application Architect** has access to the **Business IT Alignment**, **Catalog** and **Application Architecture** panes.



The Business IT Alignment pane

The **Business IT Alignment** pane provides access to **Logical Architecture** menu, to describe the logical architecture elements.

For more details on logical architectures, see ["Modeling Business Capabilities and the logical application architecture"](#), page 61.

The Application Architecture pane

The **Application Architecture** pane provides access to the **Functional Architecture** menu to describe the application systems of the information system.

For more details on functional and technical architectures, see ["Modeling Technical and Functional Architectures"](#), page 81.

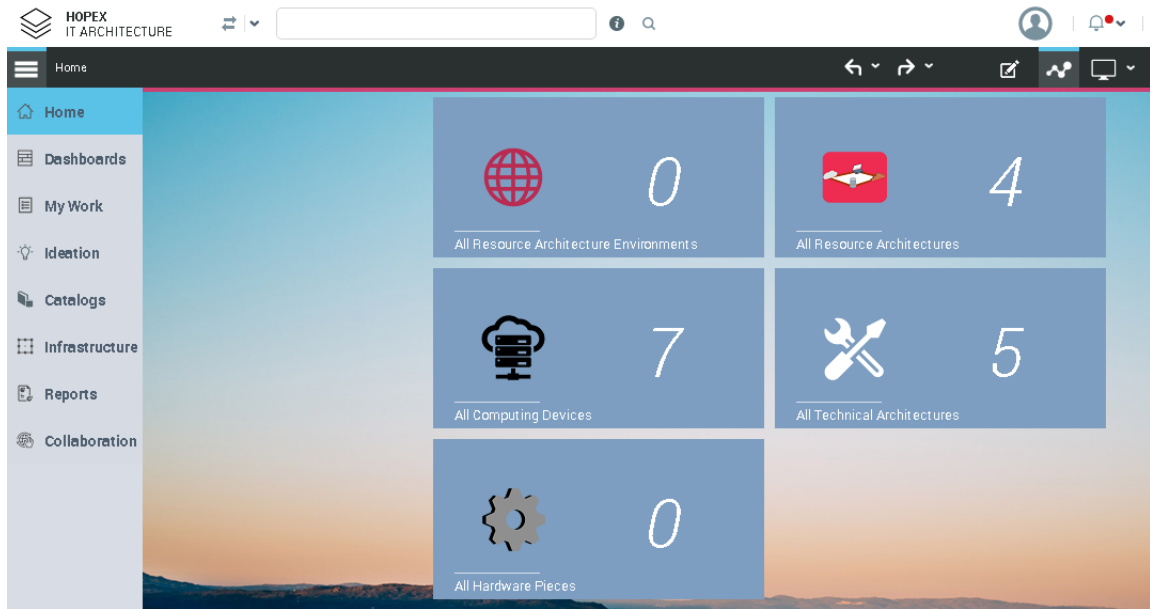
The Catalogs pane

The **Catalogs** pane provides access to the service catalogs.

For more details on service catalogs, see ["Using service catalogs"](#), page 127.

Presentation of the Infrastructure Architect space

In addition to the panes offered in standard mode to all **HOPEX IT Architecture** desktop users, the **IT Infrastructure Architect** has access to the **Catalogs** and **Infrastructure** panes.



The Catalogs pane

The **Catalogs** pane provides access to the service catalogs.

➡ For more details on service catalogs, see ["Using service catalogs", page 127](#).

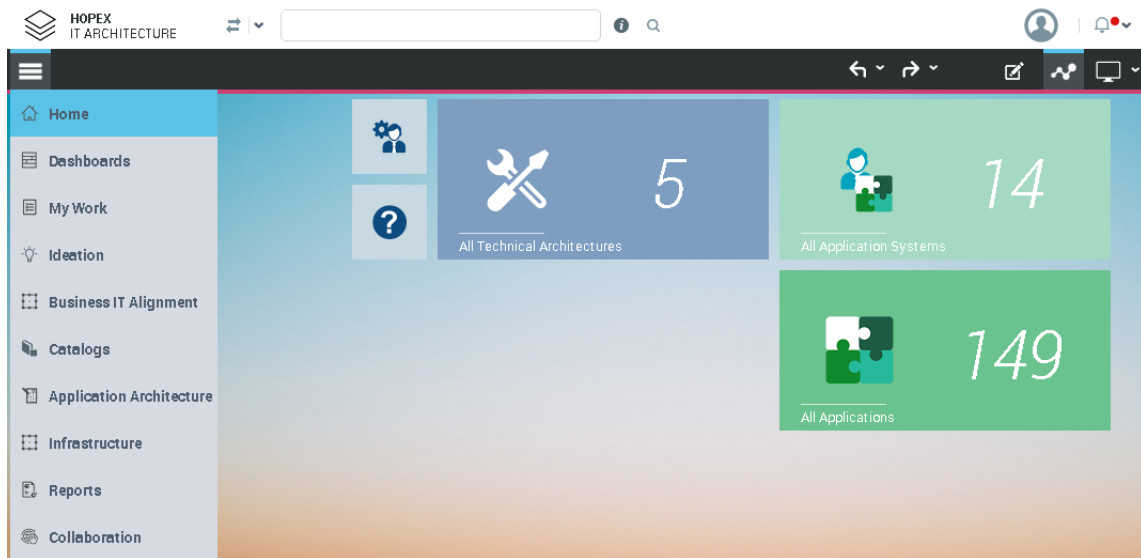
The infrastructure pane

The **Infrastructure** pane provides access to the following menus.

- **Infrastructure** to describe the elements that make up the resource architecture.
➡ For more details on infrastructures, see ["Modeling IT Infrastructures", page 111](#).
- **Technical Architecture** to describe the applications and the technology stacks of the information system.
➡ For more details on functional and technical architectures, see ["Modeling Technical and Functional Architectures", page 81](#).

Presentation of the IT Technical Architect space

In addition to the panes offered in standard mode to all **HOPEX IT Architecture** desktop users, the **IT Technical Architect** has access to the **Catalogs** and **Application Architecture** panes.



The Application Architecture pane

The **Application Architecture** pane provides access to the **Functional Architecture** menu to describe the application systems of the information system.

☛ For more details on functional and technical architectures, see ["Modeling Technical and Functional Architectures", page 81.](#)

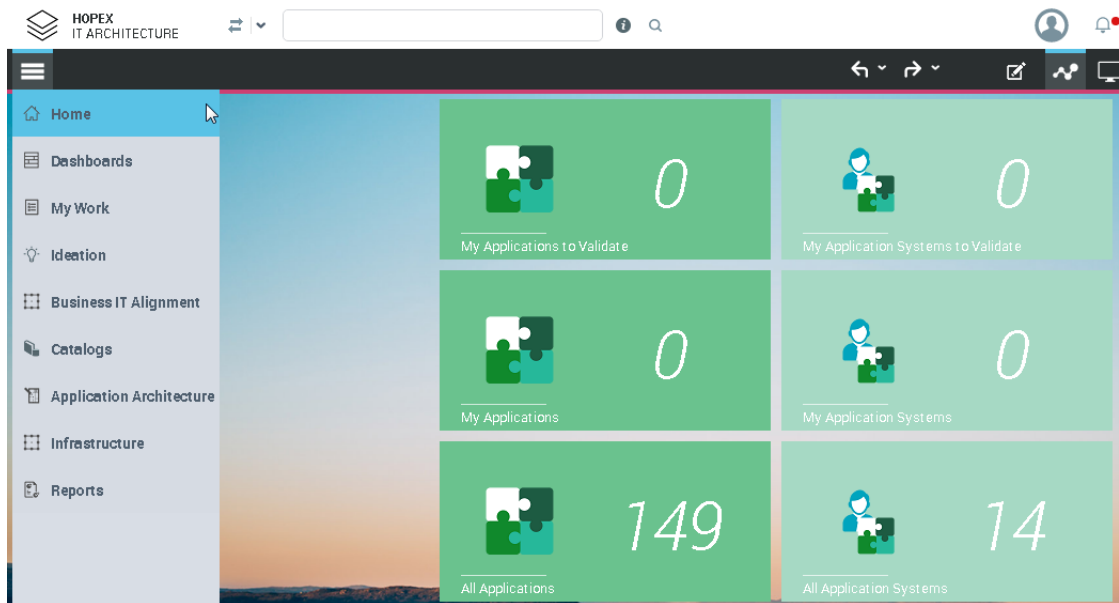
The Catalogs pane

The **Catalogs** pane provides access to the service catalogs.

☛ For more details on service catalogs, see ["Using service catalogs", page 127.](#)

Presentation of the Application Contributor space

Users connected with the **Applications Contributor** profile have access to the panes offered in standard mode to all users of the **HOPEX IT Architecture** desktop.

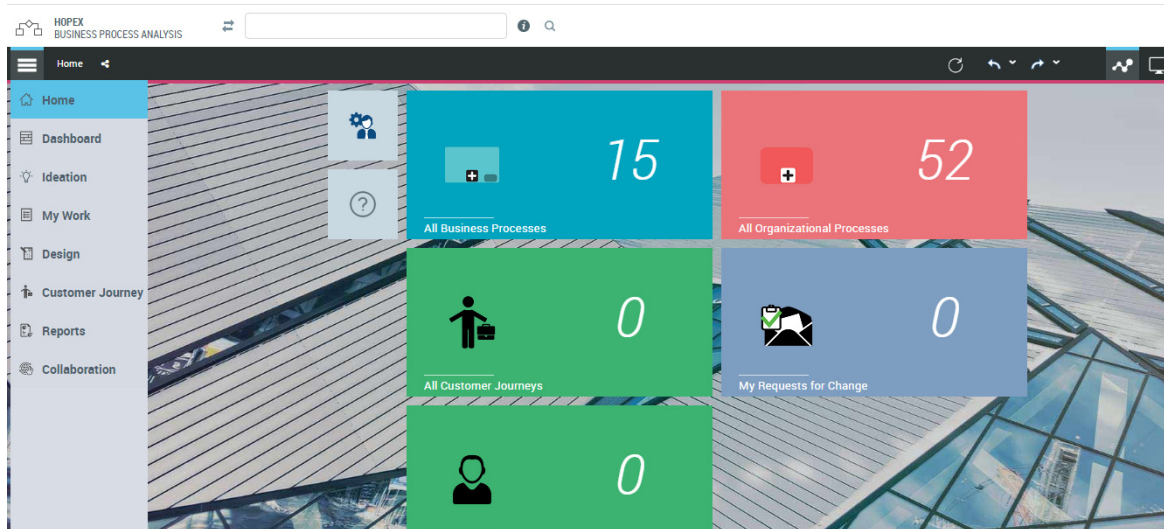


Users connected with the **Applications Contributor (lite)** profile have access to the **Enterprise Architecture** desktop for **HOPEX** solutions via the **HOPEX Explorer** application.

Presentation of the Application Viewer space

Users connected with the **Applications Viewer** profile have access to the panes offered in standard mode to all users of the **HOPEX IT Architecture** desktop.

Users connected with the **Application Viewer (Lite)** profile have access to the **Enterprise Architecture** desktop for **HOPEX** solutions via the **HOPEX Explorer** application.



Switching Between Profiles

Using the **HOPEX IT Architecture** desktop, you can access to any **HOPEX** solution desktop, without logging out, just by switching to another profile.

For example, you can switch to a specific profile:

1. Select **Main Menu > Switch Profile**.
2. Select the profile with which you want to connect.
3. (If you made modifications in your private workspace) Click:
 - **Yes**, to save your modifications in the repository.
 - **No**, if you do not want to save in the repository the modifications you made since your last dispatch. Modifications to your desktop are also lost.

The desktop associated with the selected profile is displayed.

🖱️ Click **Cancel** to stay in your private workspace.

BEFORE STARTING WITH HOPEX IT ARCHITECTURE

Preparing the Work Environment

In the context of the **HOPEX IT Architecture** solution, a *library* can hold all the elements of your project: processes and org-units, for example.



Libraries are collections of objects used to split repository content into several independent parts. They allow creation of virtual partitions of the repository. In particular, two objects owned by different libraries can have the same name.

An *Enterprise* is used to represent a work context.



An enterprise is a purposeful undertaking, an effort conducted by one or more organizations, aiming at delivering goods and services, in accordance with the enterprise mission in its changing environment. In the course of its development, the enterprise must adapt to its environment and establish the transformation objectives and goals to be achieved as well as the strategic action plans used to achieve these objectives. The development and achievement of the different adaptation and transformation stages can lead to a modification of the organization's boundaries. This requires the implementation of an integrated team, under the responsibility of a governing body, to involve the stakeholders in the transformation.

➤ For more details on managing libraries, see the "Enterprises and Libraries" chapter in the **HOPEX Common Features** guide.

Accessing the list of libraries with HOPEX IT Architecture

To access the list of libraries from the **Environment** navigation pane:

- Select **Standard Navigation > Standard Navigation**.
The library tree appears.

Creating a library with HOPEX IT Architecture

To create a library from the **Environment** navigation pane:

1. Select **Standard Navigation > Standard Navigation** in the navigation menu.
The library tree appears.
2. Right-click the **Library** folder and select **New > Library**.
A **Library** creation dialog box opens.
3. Specify the the name of the library.
4. If appropriate, enter the name of the **Owner**.
5. Click **OK**.
The library appears in the tree.

Accessing the list of enterprises with HOPEX IT Architecture

To access the list of enterprises from the **Environment** navigation pane:

- Select **Enterprises > Enterprises**.
The list of enterprises is displayed.

Creating an enterprise with HOPEX IT Architecture

To create an enterprise from the **Environment** navigation pane:

1. Select **Enterprises > Enterprises** in the navigation menu.
The list of enterprises is displayed.
2. Click the **New** button.
An enterprise creation dialog box opens.
3. Specify the name of the enterprise.
4. Select the **IT Transformation** check box.
5. Also specify the types of **Sub Containers**.
6. Click **Next**.
7. Enter the name of the **Work Environment** that you want to create and specify its **IT Architecture (Private Workspace)**.
8. Click **OK**.
The tree for the steps of a **IT Architecture (Private Workspace)** type is created automatically.
9. Click **OK**.

☛ For more details on the working environments, see the "Managing an enterprise" chapter from the **HOPEX Common Features** guide.

Choosing a Working Environment with HOPEX IT Architecture

If you work in the context of a work environment, the lists will distinguish between objects **held** by the environment and objects **imported** into the environment.



☛ To change your working environment, see chapter "Choosing a Working Environment" in the **HOPEX Common Features** guide.

Switching Between Profiles with HOPEX IT Architecture

To switch between profiles with the **HOPEX** solutions, see ["Switching Between Profiles", page 31](#).


Using properties pages

The structure of the property pages of the objects used in the **HOPEX IT Architecture** solution is often the same. This section introduces the most common pages.

Displaying the properties window on a permanent basis

You can choose to display the property windows in **HOPEX** on a permanent basis so as to view immediately the properties of an object.

To display the properties window on a permanent basis:

1. Click the **Properties**  button on the top right-hand side.
The **Properties** window appears in the Edit Area.
2. Select an object.
Its properties appear.

HOPEX IT Architecture properties pages content

HOPEX IT Architecture provides properties pages available for several solutions.

☛ *Using the facilities described in the **HOPEX Power Studio** guide, you can customizing the properties pages of your solution.*

The pages below are common to main **HOPEX IT Architecture** objects.

- the **Components** pages provide access to the list of components of the described object defined in the different diagram types.
 - the **Components > Scenario** page provides access to the list of components of the described object defined in its flow scenarios.
☛ *For more information on a scenario of flow, see ["Using a Scenario Sequence Diagram"](#), page 161.*
 - the **Components > Structure** page provides access to the list of components of the described object in its structure diagrams.
☛ *For more details on components of a structure diagram, see ["Creating an Application Structure Diagram"](#), page 156*
 - the **Components > Technical Architectures** page provides access to the list of components of the described object in its technical architecture diagrams.
☛ *For more information on the components of a technical architecture diagram, see ["Creating an Application Technical Architecture Diagram"](#), page 164.*
- the **Use** pages provide access to the information relating to the use contexts of the the described object in different types of diagrams.
 - the **Usage > Scenario** page provides access to the list of elements that use the described object in their flow scenarios.
☛ *For more information on a scenario of flow, see ["Using a Scenario of Application Flows Diagram"](#), page 158.*
 - the **Usage > Structure** page provides access to the list of elements that use the described object in their structure diagram.
☛ *For more information on the components of an application structure diagram, see ["Creating an Application Structure Diagram"](#), page 156.*
 - the **Usage > Impact Report (Structure)** page is used to analyze the impact of the failure of the described object on the elements that use it or on the applications or application systems with which it interacts via exchange contracts as described in the structure diagrams that use the described object.
☛ *For more details, see ["Impact report \(Structure\)"](#), page 194.*
 - the **Usage > Impact Report (Scenario)** page is used to analyze the impact of the failure of the described object on the application systems, applications or application services with which it exchanges

application flows in the context of the scenario of use of the described object.

☛ For more details, see *"Impact report (Scenario)", page 192.*

- The **KPI Dimension** page provides access to performance indicators connected to the described object.
☛ For more details on managing key indicators, see the *"Using KPIs" chapter in the HOPEX Business Architecture guide.*
- the **Performed Process** page provides access to the application processes executed by the described object.
☛ For more details on system processes, see *"Describing Application Processes", page 94.*
- the **Service and Request Points** page specifies the services expected or delivered by the described object.
☛ For more details, see *"Describing Service and Request Points", page 169.*
- the **Assignment** page is used to specify the managers of the described object.
- The **Reports** page provides access to the reports available for the described object.
☛ For more information on **HOPEX IT Architecture** reports, see *"HOPEX IT Architecture Reports", page 181.*

Importing components with HOPEX IT Architecture

HOPEX IT Architecture uses Excel data exchange wizards to export import and export existing architecture components.

☛ For more details on Excel data exchange wizards, see the *"Exchanging Data with Excel" chapter in the HOPEX Common Features guide.*



Two Excel templates are proposed:

- **Hardware_Hardware_Functionalities_Import_Template.xlsx** for infrastructure elements import/export : computing devices connected to the breakdown of hardware functionalities they implement.
- **Technologies_Technical_Functionalities_Import_Template.xlsx** for technology elements import/export : technologies connected on one side to vendors, and on the other side to the breakdown of technical functionalities they implement.

Structure of the import/export Excel templates of HOPEX IT Architecture

HOPEX IT Architecture Excel templates that enable import of hardware or technical elements are identically presented.

- At the level hardware, the elements are as follows: *Hardware, IoT Device, Server* and *It Device*.



An IoT device is both a hardware device and a computing device which provides combined hardware and information services to the users using it directly. As a hardware device, it embeds sensors - e.g.

accelerometer - which provide data to the embedded computing device. As a computing device, it can host data stores or run applications. Examples: smartwatch with GPS tracker, on-line surveillance video camera with live IP video feed, connected weighting scale with weight history management

☛ For more details on hardware elements, see "[Describing a Computing Device](#)", page 118.

- At the level of technologies, the elements are as follows: **Vendor (Org-Unit)** and **Software technology**.

📖 A software technology is a basic component necessary for operation of business applications. Software technologies include all basic software such as: application server, electronic mail server, software components for presentation, data entry, storage, business information sharing, operating systems, middleware, navigators, etc.

☛ For more details on technologies, see "[Describing a Software Technology](#)", page 105.

- At the level of functionalities(technical or material), the elements are as follows:

- **Technical** or **Hardware functionalities**,

📖 A functionality is a service required by an org-unit in order to perform its work. This functionality is generally necessary within an activity in order to execute a specific operation. If it is a software functionality, it can be provided by an application.

- **Technical** or **Hardware functionality maps**,

📖 A functionality map is a set of functionalities with their dependencies that, jointly, define the scope of a hardware or software architecture.

☛ For more details on functionalities, see "[Using Functionalities with HOPEX IT Architecture](#)", page 67.

- **Sub-functionalities**, which define the link between a functionality and the functionality map (or the functionality) in which it is referenced.
- **Functionality fulfillments**, which define the link between a functionality and the hardware or technical object that implements it.

The list of information provided for in the Excel template delivered with **HOPEX IT Architecture** is presented in the following order:

- For elements of type: *Hardware* or *technical functionality*, *Hardware* or *technical functionality map*, *Computing device (IoT device, Server, It device)* :
 - **Short Name** : name of the object concerned.
 - **Comment** : object comment.
- For elements of type *Technology*:
 - **Short Name** : name of the object concerned.
 - **Technology Code**.
 - **Comment** : object comment.
 - **Vendor**.
- For elements of type *Org Unit* :
 - **Short Name** : name of the object concerned.
 - **Internal/External**.
 - **Org-Unit Type**.
 - **Comment** : object comment.
- For each element of *Functionality Composition* type:
 - Name of the composite object: functionality map or functionality,
 - Name of the owned functionality.
- For each element of *Functionality Implementation* type:
 - **Fulfilled Hardware/Technical Functionality**: name of the implemented functionality.
 - Name of the object (computing device or technology) that implements the functionality.

Importing computing devices or technologies with Excel

☛ For more information on the structure of the Excel template, see *"Building the import file for HOPEX IT Architecture", page 53.*

Several steps must be followed in order for the Excel import to be performed correctly:

1. *"Checking the Excel import/export options", page 51,*
2. (optional) *"Specifying the current library", page 51,*
3. *"Importing objects in a repository", page 52.*

☛ For more information on the structure of the Excel template to be imported, see *"Building the import file for HOPEX IT Architecture", page 53.*

Checking the Excel import/export options


To be able to use the Excel import/export features:

1. Open the options window, select folder **Data Exchange > Import/Export Synchronization > Tools/Third Party Formats**.
2. Check that the option **Export Excel: Availability in Listviews** is selected.

Specifying the current library

This optional stage enables to connect imported objects to the current library.

A *library* and an *enterprise* are used to represent a unique work context.

 Libraries are collections of objects used to split repository content into several independent parts. They allow creation of virtual partitions of the repository. In particular, two objects owned by different libraries can have the same name.


In order for the data you import with Excel to be linked to a specific container, you must specify the current library.

 For more details, see ["Preparing the Work Environment"](#), page 46.

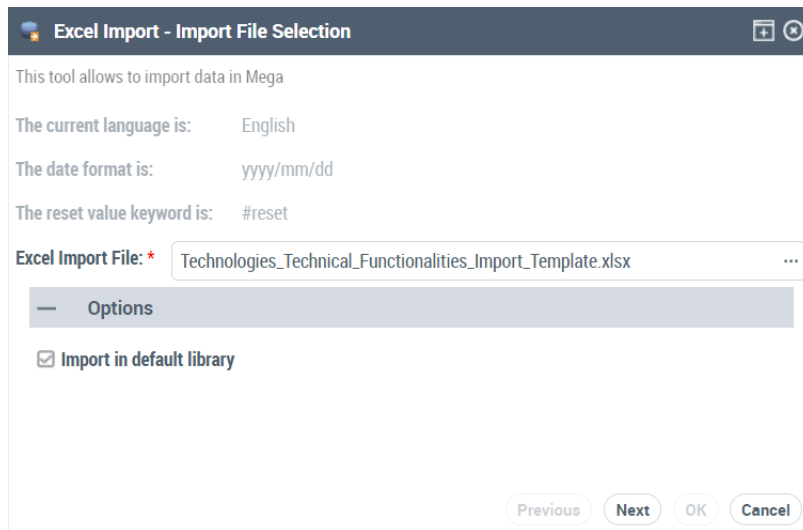
Importing objects in a repository

To import objects using the Excel file of **HOPEX IT Architecture**:

1. Click the Main menu and select **Import > Excel (*.xls, *.xlsx)**. The import wizard appears in the edit window.
2. At the right of the **Excel Import File** field, click the **Browse** button.
3. Select the file to be imported.

 For more information on the creation of the Excel file to be imported, see ["Building the import file for HOPEX IT Architecture"](#), page 53.


4. (Optional) Select the **Import in the default library** check box.



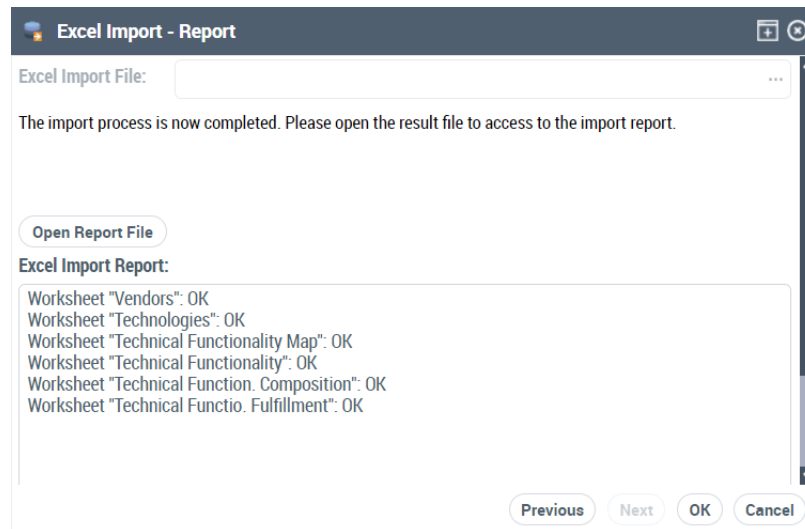
The dialog box titled "Excel Import - Import File Selection" contains the following elements:

- A header bar with a plus icon and a close icon.
- Text: "This tool allows to import data in Mega".
- Field: "The current language is:" with the value "English".
- Field: "The date format is:" with the value "yyyy/mm/dd".
- Field: "The reset value keyword is:" with the value "#reset".
- Field: "Excel Import File: *" with the value "Technologies_Technical_Functionalities_Import_Template.xlsx" and a browse button (three dots).
- A section titled "Options" with a minus icon.
- Check box: "Import in default library" (checked).
- Buttons at the bottom: "Previous", "Next", "OK", and "Cancel".

5. Click **Next**.
The list of sheets in the imported Excel table appears.

 If you select a worksheet, the list of imported fields appears in the **Worksheet Columns** section.

6. Click **Next**.
The wizard provides a report of import results.



7. To obtain a detailed report of import errors, click the **Open Report** button.
The .xls (or .xlsx) file opens indicating in color red the problem data.
8. To have the data imported into the current library, click **OK**.
9. To modify the imported file or the import parameters, click **Previous**.
10. To discard import, click **Cancel**.

Building the import file for HOPEX IT Architecture

For more information on the structure of the Excel template, see ["Structure of the import/export Excel templates of HOPEX IT Architecture", page 49](#).

If you want to export computing devices or technologies or functionality maps that exist in another repository than your current one, for example, you can use the Excel template of **HOPEX IT Architecture**.

For more details on exporting data, see ["Exporting components with HOPEX IT Architecture", page 53](#).

When the Excel file is filled with the names of the objects you want to import, you must complete the necessary information for import into **HOPEX IT Architecture**.

For more details on additional information, see ["Completing the import file for HOPEX IT Architecture", page 55](#).

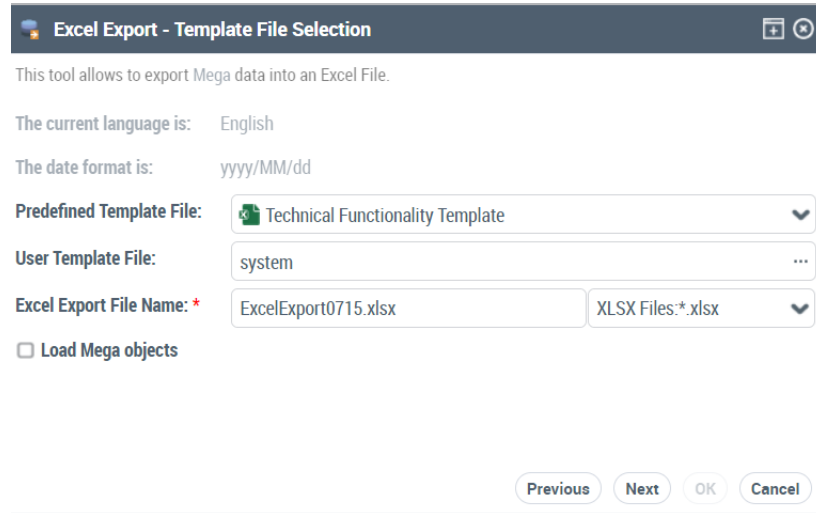
Exporting components with HOPEX IT Architecture

To access the settings of the data export wizard from **HOPEX IT Architecture** to an Excel file:

1. Check that your export options are correct. See ["Checking the Excel import/export options", page 51](#).
2. Click the Main menu and select **Import > Excel (*.xls, *.xlsx)**.
The export wizard appears in the edit window.

3. Select **From a template**.
4. Click **Next**.
5. In the filed **Predefined Template File** select **Technologies Functionality Template** if you want to import on the technology Excel template.

☛ Select **Hardware Functionality Template** if you want to import on the hardware Excel template.

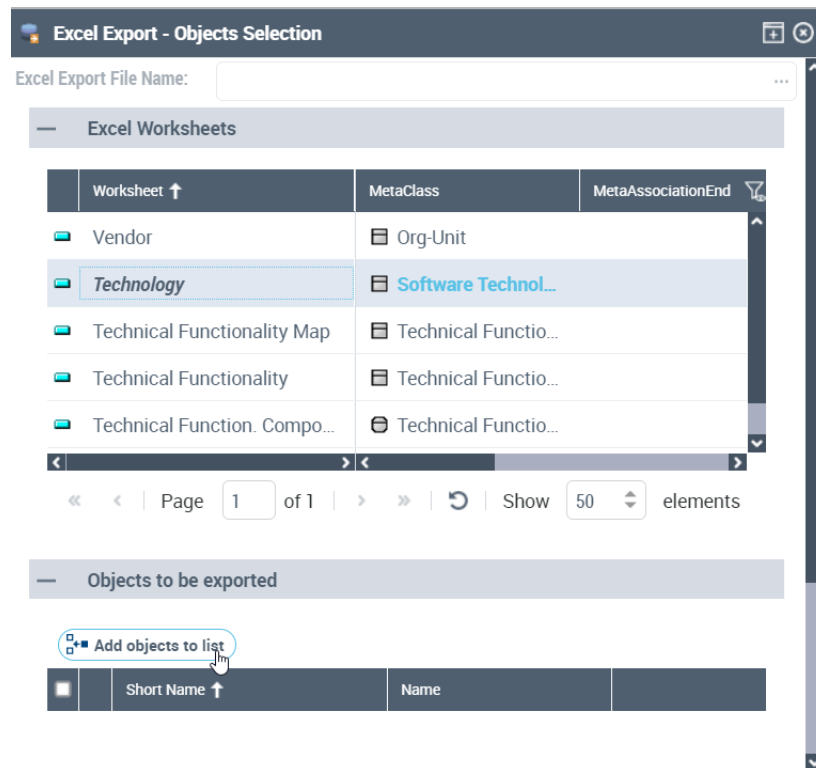


The dialog box titled "Excel Export - Template File Selection" contains the following fields and controls:

- A description: "This tool allows to export Mega data into an Excel File."
- "The current language is:" followed by a text field containing "English".
- "The date format is:" followed by a text field containing "yyyy/MM/dd".
- "Predefined Template File:" followed by a dropdown menu showing "Technical Functionality Template".
- "User Template File:" followed by a text field containing "system" and a browse button "...".
- "Excel Export File Name: *" followed by a text field containing "ExcelExport0715.xlsx" and a file type dropdown showing "XLSX Files:*.xlsx".
- A checkbox labeled "Load Mega objects" which is currently unchecked.
- At the bottom right, four buttons: "Previous", "Next", "OK", and "Cancel".

6. Click **Next**.
Export window appears to select the objects to be exported according to their type.

7. In the **Excel Worksheets** section, select the type of object you want to export and, in the **Objects to be exported** section, click **Add objects to list**.



8. From the query window, select the objects you wish to export.
9. When you have selected all the objects you want to export, click **Next**.
10. Click **Open the Excel file** to view the export file.
The file opens in an xlsx table. You can save it if you wish.
11. To modify export parameters, click **Previous**.
12. To discard export, click **Cancel**.
13. Click **OK** to finish.
The generated xlsx file is in the format expected for later import.

Completing the import file for HOPEX IT Architecture

For your import/export file to be correct, you must have specified the following elements:

- For each element of *Hardware* or *technical functionality*, *Hardware* or *technical functionality map*, *Vendor*, *Technology*, *Computing device* (It

devicee, *Server*, *IoT device*) type, you must enter the name of each object.

- For each breakdown (**Technical Function_Composion** or **Functionality Composition** Excel sheet), you must indicate:
 - The name of the composite object: functionality map or functionality,
 - The name of the owned functionality.
- To specify that a technology implements a functionality for example, you must indicate in the **Technical Function_Fulfillment** sheet:
 - the name of the functionality implemented in the **Fulfilled Hardware/Technical Functionality** column.
 - Name of the object (computing device or technology) that implements the functionality.



The first two lines of each Excel worksheet are reserved for file configuration; ensure that the first two lines of the imported file remain identical to those obtained after an export.

ABOUT THIS GUIDE

This guide explains how to make best use of **HOPEX IT Architecture** to ensure efficient management of IT Architecture.


Guide Structure

The **HOPEX Business Architecture** guide comprises the following chapters:

- ["Modeling Technical and Functional Architectures", page 81](#) ; presents the functionalities offered by HOPEX IT Architecture to describe the IT components of your enterprise.
- ["Modeling IT Infrastructures", page 111](#) ; describes the functionalities offered by **HOPEX IT Architecture** to take into account systems using resources other than software..
- ["Modeling Business Capabilities and the logical application architecture", page 61](#) ; explains how HOPEX IT Architecture helps you in analyzing your Logical Architecture.
- ["Managing IT transformation", page 137](#) ; presents how to identify and plan the transformation stages of your IS.
- ["Using service catalogs", page 127](#); explains how drawing up the list of elements that implement the functionalities of your SI.

Additional Resources

This guide is supplemented by:

- the **HOPEX Common Features** guide describes the Web interface and tools specific to **HOPEX** solutions.
 *It can be useful to consult this guide for a general presentation of the interface.*
- The **HOPEX Business Process Analysis** guide, which describes the functionalities proposed to manage processes;
- The **HOPEX Business Architecture** guide describes the functionalities proposed to manage your Business Architecture projects;;
- The **HOPEX IT Portfolio Management** guide, which describes functions proposed to manage all your applications;
- The **HOPEX Assessment** guide, which describes functions proposed by **HOPEX** to use and customize assessment questionnaires.
- the **HOPEX Power Supervisor** administration guide.

Conventions used in the guide

👉 *Remark on the preceding points.*

📖 *Definition of terms used.*

😊 *A tip that may simplify things.*

🦖 *Compatibility with previous versions.*

💣 **Things you must not do.**



Very important remark to avoid errors during an operation.

Commands are presented as seen here: **File > Open**.

Names of products and technical modules are presented in bold as seen here:
HOPEX.

MODELING BUSINESS CAPABILITIES AND THE LOGICAL APPLICATION ARCHITECTURE



The goal of this step, on a strategic level, is to check the suitability between the *business capabilities* of the enterprise and the logical architecture elements that deliver them.

This consists of the following tasks:

- ✓ "Using Business Capabilities with HOPEX IT Architecture", page 62,
- ✓ "Using Functionalities with HOPEX IT Architecture", page 67,
- ✓ "Describing a Logical Application Architecture with HOPEX IT Architecture", page 71.

USING BUSINESS CAPABILITIES WITH HOPEX IT ARCHITECTURE

Business capabilities examples with HOPEX IT Architecture

A *business capability* defines an expected skill.



A business capability is a set of features that can be made available by a system (an enterprise or an automated system).

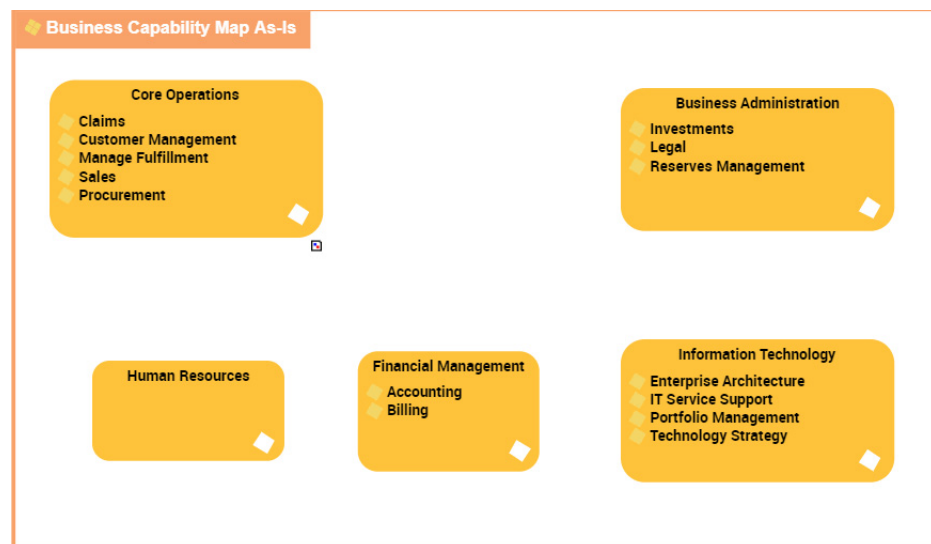
For example, to respond to a customer satisfaction objective, the organization must be able to provide services conforming to contractual commitments.

A *business capability map* describes what the enterprise is capable of producing for its internal needs or for meeting the needs of its clients. It is thus based on the main business capabilities of its activity at a given moment.



A business capability map is a set of business capabilities with their dependencies that, together, define a framework for an enterprise stage.

For example, the standard ability to manage "Operational Activities" is based on the business capabilities to process "Supply", "Sales" and "Complaints", "Order Management" and "Customer Management".




*For more details on managing a business capability map, see the "Describing a business capabilities map" chapter in the **HOPEX Business Architecture** guide.*

The description of *business capabilities* and *functionalities* is particularly interesting if business capabilities are associated with the functionalities that fulfill them.

Furthermore, if *applications* are connected to the *functionalities* they implement, they are indirectly connected to *business capabilities*. In **HOPEX IT Strategy**, a report allows to check the functional coverage of your applications.

☛ For more details on the business capabilities reports, see "[Business Capabilities reports](#)", page 182.

Managing the Business Capability Maps with HOPEX IT Architecture

 A business capability map is a set of business capabilities with their dependencies that, together, define a framework for an enterprise stage.

Accessing the list of business capability maps

To access the list of capabilities maps from the **Business IT Alignment** navigation pane:

1. Select **Logical Architecture** in the navigation menu.
2. Click on **Business Capability Map** tile
The list of Capabilities maps appears in the edit area.

Creating a business capability map

To create a business capability map:

1. Select **Business IT Alignment > Logical Architecture** in the navigation menu.
2. Click on **Business Capability Map** tile
The list of Capabilities maps appears in the edit area.
3. Click the **New** button.
4. In the creation dialog box, enter the name of the business capabilities map and click **OK**.
The new business capability map is added to the list of existing capability maps.

Creating a business capability map diagram

To create a business capability map diagram:

1. Right-click the business capability map that interests you and select **New > Business Capability Map Diagram**.
The diagram opens in the edit area. The frame of the business capability map described appears in the diagram.
You can construct this diagram in tabular input mode.


The properties of a business capability map

The **Characteristics** property page of Capabilities map provides access to:

- its **Owner**, by default during creation of the object, the current enterprise.
- its **Name**,
- the text of its **Description**.

With **HOPEX IT Architecture** a business capability is described by the following pages:

- the **Structure** page that specifies the list of business capability map components owned and the dependencies between them.

 For more details on business capability map components, see ["Creating a business capability map", page 63](#).

- The **Capability usage** page, which is used to identify the IT transformation stages that use this functionality map.
- the **Assignment** page, which is used to specify the managers of the capability map.

 For more details on other property pages proposed by **HOPEX IT Architecture**, see ["HOPEX IT Architecture properties pages content", page 47](#).

Managing Business Capabilities with HOPEX IT Architecture




.A business capability is a set of features that can be made available by a system (an enterprise or an automated system).

Describing a business capability

A business capability is described in more detail by the following elements:

- a more detailed granularity capability breakdown;
- the expected effects of the capability;
- The required functionalities, see ["Defining the functionalities associated with business capabilities", page 65](#);
- the dependencies between capabilities (expected effect of one dependent from the result of the other).

 For more details on managing a business capability, see the ["Describing a business capability" chapter in the HOPEX Business Architecture guide](#).

For example, the business capability that consists of taking "Customer Call" is broken down into a number of business capabilities: "Call analysis", "Identification of the customer".


Accessing the list of business capabilities with HOPEX IT Architecture

To access the list of capabilities from the **Business IT Alignment** navigation pane:

1. Select **Logical Architecture** in the navigation menu.

2. Click on **Business Capability** tile.
The list of business capabilities appears in the edit area.

Defining the functionalities associated with business capabilities

 A *functionality* is a service required by an org-unit in order to perform its work. This functionality is generally necessary within an activity in order to execute a specific operation. If it is a software functionality, it can be provided by an application.

Each business capability is associated with functionalities that it is able to provide and that it needs to ensure its functionalities.

For example, the "Customer Management" needs the "get customer information" functionality.

➤ For more information on enterprise functionalities, see ["Describing functionalities with HOPEX IT Architecture", page 68](#).

To associate a *functionality* with a business capability:

1. Open the property pages of the business capability concerned and select the **Expected Capabilities** page.
2. In the **Expected Functionality** section, click **New**.
The **Expected Functionality** creation dialog box opens.
3. Click, for example, the **Creating a New Functionality** check box.
4. Specify the name of the functionality.
5. Click **OK**.
The expected functionality appears in the list of functionalities associated with the business capability.

The functionalities and the expected effects appear in the diagrams, at the bottom of the frame of the capability described.

➤ For more information on enterprise functionalities, see ["Describing functionalities with HOPEX IT Architecture", page 68](#).

Describing implementation of a business capability


This involves connecting the *business capability*, which corresponds to what we know how to do or what we want to do and which represents the goal to be achieved, to the means of implementation represented by *applications* (or *logical applications* at a conceptual level) or *application systems* (or *logical application systems*).

➤ Conceptual representations are made before organizational and technical choices are made.

Creating a business capability realization

A business capability can be achieved either by an application or application system, or at a conceptual level, by a logical application or application system.

To associate an application with a business capability, you must create a business capability map realization.

 The creation of a business map represents the organization of physical agents (Application Systems) or logical (Business Function) agents that implements the business capacities of the map.

To specify that a business capability is fulfilled by an existing application:

1. Open the **Implementation** property page of the business capability that interests you.
2. Click **New**.
The creation window for a business capability realization opens.
3. Select **Reusing an existing....**
4. Select the **Application** object type.
5. Select the application that interests you and click **OK**.
The capability realization appears in the list with the name of the selected application.


Analyzing enterprise capability implementation

HOPEX IT Architecture provides reports to display realization coverage of business capability elements by operational elements such as applications, and according to different perspectives: Organizational, Business/Data, Logical/Physical Application, etc.


☛ *For more details on fulfillment reports for enterprise capabilities, see ["Business Capabilities reports"](#), page 182.*

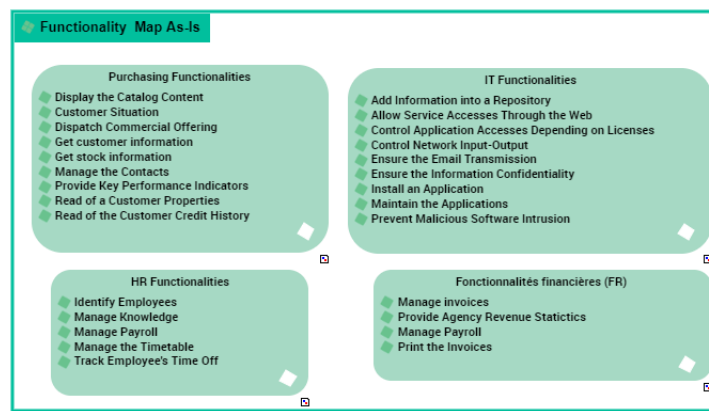
USING FUNCTIONALITIES WITH HOPEX IT ARCHITECTURE

A **functionality** is an aptitude expected from an equipment.


 A functionality is a service required by an org-unit in order to perform its work. This functionality is generally necessary within an activity in order to execute a specific operation. If it is a software functionality, it can be provided by an application.

A **functionality map** describes all the functionalities the enterprise is able to cover for its internal needs or for meeting the needs of its clients.


 A functionality map is a set of functionalities with their dependencies that, jointly, define the scope of a hardware or software architecture.



Example of a functionality map

 For more details on managing a functionality map, see the "Describing the Functionality Map" chapter in the **HOPEX Business Architecture** guide.

Describing a Functionality Map with HOPEX IT Architecture

 A functionality map is a set of functionalities with their dependencies that, jointly, define the scope of a hardware or software architecture.

Accessing the list of functionality maps with HOPEX IT Architecture

To access the list of capabilities from the **Business IT Alignment** navigation pane:



1. Select **Logical Architecture** in the navigation menu.
2. Click **Functionality Maps** in the navigation menu.
The list of functionality maps appears in the edit area.

The properties of a functionality map

The **Characteristics** properties page of a functionality map provides access to:

- its **Owner**, by default, when creating the enterprise or business capability map, this is the current library.
- its **Name**,
- the text of its **Description**.

With **HOPEX IT Architecture**, a functionality map is described by the following pages:

- the **Structure** page is used to specify a list of components owned and the dependencies between them.
 For more information on the components of a functionality map, see ["Creating a Functionality Diagram with HOPEX IT Architecture"](#), page 69.
- the **Implementation** property page is used to specify the environments that make it possible to create the described functionality map.
 For more details on implementation of functionalities, see ["Creating a Functionality Realization"](#), page 69.
- The **Usage** page, which is used to identify the IT transformation stages that use this functionality map.
- the **Assignment** page, which is used to specify the managers of the functionality map.

Creating a functionality map

To create a functionality map diagram:

- 1 Right-click the functionality map that interests you and select **New > Functionality Map Diagram**.

The diagram opens in the edit area. The frame of the functionality map described appears in the diagram.

To create a functionality in a functionality diagram, see "Creating a functionality component in a functionality map diagram" chapter in **HOPEX Business Architecture** guide.

To define the dependencies of sub-functionalities, see "Defining Functionality dependencies" chapter in **HOPEX Business Architecture** guide.

Describing functionalities with HOPEX IT Architecture



A functionality is a service required by an org-unit in order to perform its work. This functionality is generally necessary within an activity in order to execute a specific operation. If it is a software functionality, it can be provided by an application.


To access the list of functionalities using the **Logical Application Architecture** navigation pane:

- 1 Select **Logical Application Architecture Inventories > Functionalities** in the navigation menu.

The list of functionalities appears in the edit area.

The **Characteristics** property page of the functionality provides access to:

- its **Owner**, by default during creation of the functionality, the current enterprise.
- its **Name**,
- the text of its **Description**.
- the **Desired Application Effects**:

 For more information on the effects of expected functionalities, see "Creating a Functionality Diagram with HOPEX IT Architecture", page 69.

Creating a Functionality Diagram with HOPEX IT Architecture

To create a functionality diagram:

1. Right-click the functionality that interests you and click **New > Functionality diagram**.


The diagram opens in the edit area. The frame of the functionality described appears in the diagram.

To create a functionality in a functionality diagram, see "Creating a functionality component in a functionality map diagram" chapter in **HOPEX Business Architecture** guide.

To define the dependencies of sub-functionalities, see "Defining Functionality dependencies" chapter in **HOPEX Business Architecture** guide.

Describing Implementation of a Functionality


This involves connecting the *functionality*, which contributes to the goal to be achieved, to the means of implementation represented by *applications* (or *logical applications* at a conceptual level) or *application systems* (or *logical application systems*).

 Conceptual representations are made before organizational and technical choices are made.

Creating a Functionality Realization

A functionality can be achieved either by an application or application system, or at a conceptual level, by a logical application or application system.

To associate an application with a functionality, you must create a functionality realization.

 A realization describes the relationship between a logical entity and a physical entity that implements it. The physical entity gives the list of logical entities that it implements.

To specify that a functionality is implemented by an application:

1. Open the **Implementation** property page of the functionality that interests you.
2. Click **New**.
The creation window for a functionality realization opens.
3. Select **Reusing an existing....**
4. Select the **Application** object type.

5. Select the application that interests you and click **OK**.
The functionality realization appears in the list with the name of the selected application.

Analyzing functionality implementation

HOPEX IT Architecture provides reports to display realization coverage of functionalities by operational elements such as logical or physical application components:

➤ *For more details on fulfillment reports for enterprise capabilities, see ["Analyzing the Functional Coverage of the Architecture Implemented"](#), page 32.*

DESCRIBING A LOGICAL APPLICATION ARCHITECTURE WITH HOPEX IT ARCHITECTURE

Describing a Logical Application System with HOPEX IT Architecture

A project for describing the logical architecture of an information system inventories the existing *logical application systems* and their interactions.



A logical application system is an assembly of other application architectures, logical applications and end users, interacting with application components to implement one or several functions.

A *logical application system* is described by a structure diagram of the logical application system that represents the different components of the application systems and their interactions.

Accessing the list of logical application systems with HOPEX IT Architecture

To access the list of application systems from the **Business IT Alignment** navigation pane:

1. Select **Logical Architecture** in the navigation menu.
2. Click the **Logical Application Systems** tile.
The list of logical application systems appears.

Creating a Logical Application System

To create a *logical application system*:

1. From the **Business IT Alignment** navigation pane, select **Logical Architecture > Logical Application Systems**.
The list of application systems appears in the edit area.
2. Click **New**.
The **Creation of a Logical Application System** dialog box appears.
3. Enter the **Name** of your logical application system and click **OK**.
The new logical application system appears in the list.

Logical Application System Properties

The **Characteristics** properties page for a logical application system provides access to:

- its **Name**,
- its **Owner**, by default, during creation of the logical application system, the current library.
- the text of its **Description**.
- its **Owned Realizations** that represent the list of functionalities covered by the logical application system.

☛ For more details on functionalities, see ["Describing functionalities with HOPEX IT Architecture"](#), page 68.

With **HOPEX IT Architecture**, a logical application system is described by the following pages:

- the **Properties** page, used to specify the properties that appear in the diagrams at the bottom of the described object frame.
- the **Structure** page provides access to the list of application system components described in its different diagrams as well as the communications that exist between them.

☛ For more information on the components of a logical application system, see ["Describing a Logical Application System with HOPEX IT Architecture"](#), page 72.

- the **Implementation** page is used to specify the logical or physical elements that implement the described logical application system.

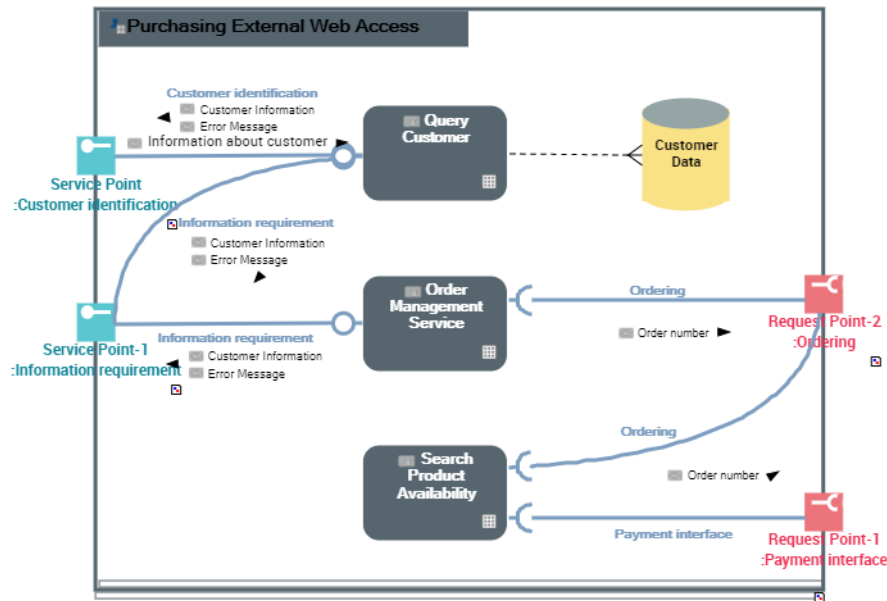
☛ For more details on other property pages proposed by **HOPEX IT Architecture**, see ["HOPEX IT Architecture properties pages content"](#), page 47.

Describing a Logical Application System with HOPEX IT Architecture

With **HOPEX IT Architecture**, the components of a logical application system and their exchanges are described in a **logical application system structure diagram**.

The logical application system structure diagram, for managing "Internet Purchase Requests", presents different

logical applications, access to a logical database as well as service and request points for "Book" or "Order".



Structure diagram of the "Internet Purchase Requests" logical application system

For more details on a logical application system representation, see ["Describing a Logical Application System with HOPEX IT Architecture", page 72.](#)

A structure diagram of the logical application system includes the following elements:

- end users**

The end user represents an organizational unit interacting at the boundaries of an application system or a logical application system.

For more details on adding end users, see ["Adding an end user to the structure diagram of the logical application system", page 74.](#)

- Logical Application System Components and Logical Application Components**

A logical application is a set of application functionalities that is independent of a particular implementation. For example, the classification of all purchase request processing applications implemented in an enterprise.

For more details on adding applications, see ["Adding a logical application to the structure diagram of the logical application system", page 74.](#)

- interactions** between the components representing requests for services

An interaction represents a contract established in a specific context between autonomous entities that are internal or external to an enterprise. These entities can be enterprise org-units, applications,

activities or processes, as well as external org-units. The content of this contract is described by an exchange contract.

☛ For more details on interactions between logical application system components, see ["Managing Interactions", page 168](#).

- **service points**



A service point is a point of exchange by which an agent offers a service to potential customers.

- **request points**



A request point is a point of exchange by which an agent requests a service from potential suppliers.

☛ For more information on access points, see ["Describing Service and Request Points", page 169](#).

Adding an end user to the structure diagram of the logical application system

To create an **end user**:

1. In the objects toolbar of the structure diagram of the logical application system, click **End User**.
2. Click in the frame of the described logical application system.
An addition window prompts you to choose the **Object Type** that you wish to use:
3. For example, select the **Org-unit** object type.



An org-unit represents a person or a group of persons that intervenes in the enterprise business processes or information system. An org-unit can be internal or external to the enterprise. An internal org-unit is an organizational element of enterprise structure such as a management, department, or job function. It is defined at a level depending on the degree of detail to be provided on the organization (see org-unit type). Example: financial management, sales management, marketing department, account manager. An external org-unit is an external entity that exchanges flows with the enterprise. Example: customer, supplier, government office.


4. Select the org-unit that interests you and click **OK**.
The actor appears in the diagram.

Adding a logical application to the structure diagram of the logical application system

To describe that a logical application system implements a logical application:

1. In the objects toolbar of the structure diagram of the logical application system, click **Logical Application component** and click in the frame of the logical application system described.
An addition dialog box prompts you to select the **Logical Application** used.
2. Select an existing logical application.
3. Click **OK**.
The logical application appears in the diagram.

Describing Logical Applications With HOPEX IT Architecture

 A logical application is a set of application functionalities that is independent of a particular implementation. For example, the classification of all purchase request processing applications implemented in an enterprise.

Accessing the list of logical applications with HOPEX IT Architecture

To access the list of logical application from the **Business IT Alignment** navigation pane:

1. Select **Logical Architecture** in the navigation menu.
2. Click the **Logical Applications** tile.
The list of logical applications appears in the edit area.

Creating a logical application

To create a *logical application*:

1. From the **Business IT Alignment** navigation pane, select **Logical Architecture > Logical Application Systems**.
The lists of logical applications appear in the edit area.
2. Click **New**.
The **Create a Logical Application** window appears.
3. Enter the **Name** of your logical application and click **OK**.
The new logical application appears in the list.

Logical Application Properties

The **Characteristics** properties page of the logical application provides access to:

- its **Name**,
- its **Owner**, by default during creation of a logical application, the current library.
- the text of its **Description**.
- its **Owned Realizations** that represent the list of functionalities covered by the logical application.

➡ For more details on functionalities, see "[Describing functionalities with HOPEX IT Architecture](#)", page 68.

With **HOPEX IT Architecture**, a logical application is described by the following pages:

- the **Properties** page, used to specify the properties that appear in the diagrams at the bottom of the described object frame.
- the **Structure** page provides access to the list of components of the logical application as well as the communications that exist between them in its different diagrams.

☛ For more information on the components of a logical application diagram, see ["Creating a structure diagram", page 156](#).

- the **Implementation** page is used to specify the logical or physical elements that implement the described logical application system.

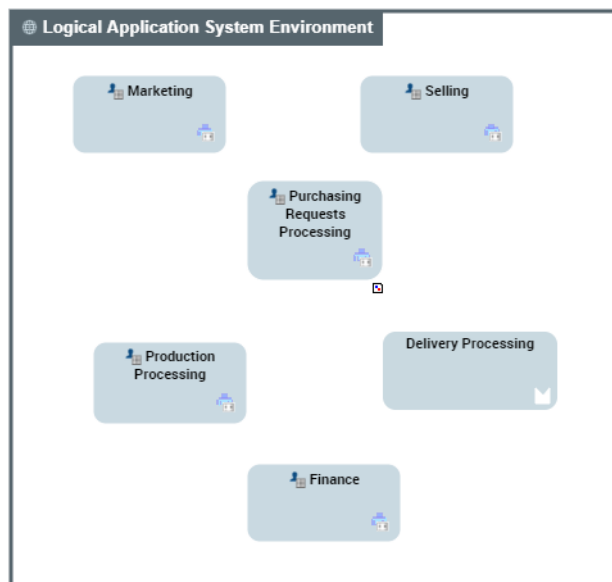
☛ For more details on other property pages proposed by **HOPEX IT Architecture**, see ["HOPEX IT Architecture properties pages content", page 47](#).

Logical Application System Environment Description

📖 A logical application system environment presents a logical application system use context. It describes the interactions between the logical application system and its external partners, which allows it to fulfill its mission and ensure the expected functionalities.

Example of logical application system environment

The internal logical application system "Purchase request processing" uses a logical "Delivery" application system that is external to the described environment.



Accessing the list of logical application system environments

To access the list of logical application system environments from the **Business IT Alignment** navigation pane:

1. Select **Logical Architecture** in the navigation menu.
2. Click the **Logical Application System Environment** tile.
The list of logical application system environments appears in the edit area.

Creating a logical application system environment

To create a *logical application system environment*:

1. From the **Business IT Alignment** navigation pane, select **Logical Architecture > Logical Application System Environments**.
The list of logical application system environments appears in the edit area.
2. Click **New**.
The **Creation of Logical Application System Environment** window appears.
3. Enter the **Name** of your application system environment and click **OK**.
The new logical application system environment appears in the list.

Logical application system environment properties

The **Characteristics** properties page for a logical application system environment provides access to:

- its **Name**,
- its **Owner**, by default during creation of a logical application system environment, the current library.
- the text of its **Description**.
- its **Owned Realizations** that represent the list of elements that the logical application system environment implements.

With **HOPEX IT Architecture**, a logical application system environment is described by the following pages:

- the **Properties** page, used to specify the properties that appear in the diagrams at the bottom of the described object frame.

➡ For more details on other property pages proposed by **HOPEX IT Architecture**, see ["HOPEX IT Architecture properties pages content"](#), page 47.

Using the Environment Structure Diagram of a Logical Application System

With **HOPEX IT Architecture**, a logical application system environment is described by a structure diagram of the logical application system environment that describes the interactions between the internal logical application systems, its users and the partner logical systems.

The diagram includes:

- **logical application systems** that represent the logical application systems internal to the described environment.

In the example, this is the logical application system
"Internet Purchasing"



A logical application system is an assembly of other application architectures, logical applications and end users, interacting with application components to implement one or several functions.

- **partner logical systems** that represent the logical application systems external to the described environment.

In the example, this is the logical application system
"Purchasing Financing"



A partner logical system is a logical application system external to the environment of the described logical application system. The partner logical system can be a service supplier or a service consumer with respect to components of the logical application system.

- **Org-Units** and **Position types** that represent the user category of services provided by the environment.

☛ See *Using Org-Units in a Diagram*.

- **Interactions** between the components representing requests for services.



An interaction represents a contract established in a specific context between autonomous entities that are internal or external to an enterprise. These entities can be enterprise org-units, applications, activities or processes, as well as external org-units. The content of this contract is described by an exchange contract.

☛ For more details, see ["Creating an Interaction", page 169](#).

Using Org-Units in a Diagram

Org-Units are used in several diagrams. The main points concerning this object type are reminded in this section.



An org-unit represents a person or a group of persons that intervenes in the enterprise business processes or information system. An org-unit can be internal or external to the enterprise. An internal org-unit is an organizational element of enterprise structure such as a management, department, or job function. It is defined at a level depending on the degree of detail to be provided on the organization (see org-unit type). Example: financial management, sales management, marketing department, account manager. An external org-unit is an external entity that exchanges flows with the enterprise. Example: customer, supplier, government office.

Creating an org-unit

To create an org-unit from the **Environment** navigation pane:

1. Select **Organization > Org-Units**.
The list of all org-units appears.
2. Click the **New** button.
3. In the **Creation of Org-Unit** dialog box, enter the name of the org-unit you want to create.

4. Click **OK**.
The org-unit appears in the list.

Internal org-unit/external entity

During creation, org-units are considered as elements internal to the company.

To specify that an org-unit is not part of the company, you must modify the org-unit properties and enter the "External" status.

To assign the "External" characteristic to the org-unit:

1. Right-click the org-unit and select **Properties**.
2. From the **Characteristics** tab, click in the **Internal/External** field and select "External" value.
3. Click on **OK** to apply the update and close the properties dialog box.
This characteristic is represented graphically and is automatically displayed in the diagrams.



MODELING TECHNICAL AND FUNCTIONAL ARCHITECTURES



HOPEX IT Architecture is used by enterprises and other organizations to represent and document their IT architectures according to a service-oriented architecture.

Modeling an architecture according to a service-oriented approach facilitates the analysis of communications between architectures. Thus, the description of architectures is based on specific concepts that enable a more generic use of the tool.

The following points are covered here:

- ✓ ["Describing Application Architecture", page 82;](#)
- ✓ ["Managing Data", page 96;](#)
- ✓ ["Describing Application Flows", page 100;](#)
- ✓ ["Describing Technical Architecture", page 105.](#)

DESCRIBING APPLICATION ARCHITECTURE

HOPEX IT Architecture Concepts Overview

The information system can be broken down according to two levels of detail: the application and the application system.

Application

An application is a set of software components constituting a coherent whole regarding deployment, functional coverage and IT techniques used.

The application is the management and deployment unit of a set of software components. An application can be deployed on one or several machines. An application meets:

- business requirements

Examples: billing, accounting, equipment management, load/capacity calculation.

- technical requirements

Examples: specific communication interface, access control.

- transverse requirements

Examples: electronic mail, directories, office system applications.

For the creation of applications, see ["Describing an Application with HOPEX IT Architecture", page 83](#).

Application System

An application system is an assembly of applications responding to a coherent set of functionalities, implemented by the applications making up the system.

An application system can comprise a suite of applications grouped for commercial reasons (integrated management software packages such as SAP, Oracle Applications, Siebel...).

An application system can also correspond to a group of applications that have the same functional objectives (accounts and financial management system integrating all accounting applications: general, suppliers, analyses, as well as financial and budgetary analysis modules, human resources management systems integrating salaries, time management, career management, etc.).

The application system, like an application, can be the subject of specific developments (carried out internally or bought-in/sub-contracted) or they can be proprietary market products (software packages).

The logical organization and structure of application systems and applications, together with description of their exchanges, constitutes the foundations of the application architecture.

For the creation of an application system, see ["Describing an Application System", page 86](#).

Describing an Application with HOPEX IT Architecture

A project for describing the functional architecture of an information system is used to inventory the existing *applications* and their interactions.



An application is a software component that can be deployed and provides users with a set of functionalities.

An *application* is described by several types of diagrams:

- An *scenario of application flows* describes the flows exchanged between the IT services or the micro-services used by this application. A scenarios can represent a particular application use case or more globally all the flows exchanged within this application.
 ➤ For more details, see ["Describing a scenario of flows", page 158](#).
- an *application structure diagram* is used to represent the interactions between the application components in the form of exchange contracts.
 ➤ For more details, see ["Creating a structure diagram", page 156](#).
- an *application technical architecture* used to represent the technical elements that support the application.
 ➤ For more details, see ["Describing a Technical Architecture", page 164](#).

Creating an Application with HOPEX IT Architecture

To create an *application*:

1. From the **Application Architecture** navigation pane, select **Functional Architecture > Applications**.
The list of applications appears in the edit area.
2. Select **My Applications** tab, for example.
3. Click **New**.
4. The **Creation of Application** dialog box appears.
5. Enter the **Name** of your application and click **OK**.
The new application appears in the list.

The properties of an application with HOPEX IT Architecture

The **Characteristics** property page of an application provides access to different sections.

- The **Identification** section provides access to the following information:
 - the **Name**
 - its **Owner**, by default during creation of the application, the current library.
 - the text of its **Description**.
 -
 - the internal **Code**
 - the **Type of application**
 - if it is an **Application template**: to be selected if the application is used to create other applications.
 - a **Comment**.
- the **business processes** that use the application.



A business process represents a system that offers products or services to an internal or external client of the company or organization. At the higher levels, a business process represents a structure and a categorization of the business. It can be broken down into other processes. The link with organizational processes will describe the real implementation of the business process in the organization. A business process can also be detailed by a functional view.

- the **Functional Scope** of the application. See ["Defining Application Functional Scope", page 85](#).
- the **Responsibility**: it relates to the person or persons responsible for the application.
 - Software Designer
 - Local Application Owner
 - IT Owner



For more details on these roles, see ["Business Roles of HOPEX IT Architecture", page 14](#).

- the **Technologies** section provides access to the list of **Technologies** and the list of **technology stacks** used by the application.



A software technology is a basic component necessary for operation of business applications. Software technologies include all basic software such as: application server, electronic mail server, software components for presentation, data entry, storage, business information sharing, operating systems, middleware, navigators, etc.



A software technology stack is a set of software technologies.



For more details on software technologies, see ["Describing a Software Technology", page 105](#).

- **Exchanges** with other objects. See ["Describing Application Flows", page 100](#).
- the **Risks** associated with the application. See ["Specifying the Risks Associated with an Application", page 85](#).
- associated **Attachments**.





*For more details on other property pages proposed by **HOPEX IT Architecture**, see ["HOPEX IT Architecture properties pages content", page 47](#).*


Defining Application Functional Scope


To indicate the objects that define application functional coverage:


1. Open the **Characteristics** property page of the application.
2. Expand the **Functional Scope** section.
The types of data that define functional coverage of the application are:
 - The *business capabilities* covered by the application


 *A business capability is a set of features that can be made available by a system (an enterprise or an automated system).*


 For more details on business capabilities, see ["Using Business Capabilities with HOPEX IT Architecture"](#), page 62.


 A report covers distribution of applications in business capabilities, see ["Business Capabilities reports"](#), page 182 .
 - The *functionalities* implemented by the application

 *A functionality is a service required by an org-unit in order to perform its work. This functionality is generally necessary within an activity in order to execute a specific operation. If it is a software functionality, it can be provided by an application.*

 For more details on functionalities, see ["Describing a Functionality Map with HOPEX IT Architecture"](#), page 67.
 - The **Logical realizations** enable to define the *logical applications* and the *logical application systems* that the application implements.

 *A logical application is a set of application functionalities that is independent of a particular implementation. For example, the classification of all purchase request processing applications implemented in an enterprise.*

 *A logical application system is an assembly of other application architectures, logical applications and end users, interacting with application components to implement one or several functions.*

 For more details on logical architectures, see ["Describing a Logical Application Architecture with HOPEX IT Architecture"](#), page 71.

Specifying the Risks Associated with an Application

HOPEX IT Architecture is used to identify the risks associated with an application, and to retrieve the evaluations defined in the **HOPEX Enterprise Risk Management** solution. You can define a new risk using the application or connect a previously defined risk.

To connect a risk to an application:

1. Open the **Characteristics** property pages of the application.
2. Expand the **Risk** section.
3. Click **Connect**.
The query dialog box appears.
4. Find and select the risk required and click **OK**.

For more details on risks and their evaluation, see **HOPEX Enterprise Risk Management**.

Describing an Application Environment with HOPEX IT Architecture



An application environment is used to represent a use context of an application. An application environment allows presenting the other application systems, applications, micro-services or actors with which this application can interact.

Accessing the List of Application Environments

To access the list of application environments from the **Application Architecture** navigation pane:

1. Select **Functional Architecture > Application Environments**.
The list of application environments appears in the edit area.

Creating an application environment

To create an *Application environment*:

1. From the **Application Architecture** navigation pane, select **Functional Architecture > Application Environments**.
The list of application environments appears in the edit area.
2. Click **New**.
The **Creation of Application System Environment** window opens.
3. Enter the **Name** of your application system environment and click **OK**.
The new application system environment appears in the list.

Application environment properties

The **Characteristics** properties page of an application environment provides access to:

- its **Owner**, by default during creation of an application system environment, the current library.
- its **Name**,
- the text of its **Description**.

With **HOPEX IT Architecture** an application environment is described by other property pages. See "[HOPEX IT Architecture properties pages content](#)", page 47.




Describing an Application System

A project for describing the functional architecture of an information system is also used to inventory the existing *application systems* and their interactions.



An application system is an assembly of other application systems, applications and end users interacting with application components to implement one or several functions.

An *application system* is described by several types of diagrams:

- a *scenario of application system flows* presents the flows exchanged between the application systems, the applications or the micro-services used by this application system. A scenario can represent a particular use case of the application system or more globally all the flows exchanged within this application system.
 For more details, see ["Describing a scenario of flows", page 158.](#)
- an *application system structure diagram*, used to represent the interactions between the application components in the form of exchange contracts.
 For more details, see ["Creating an Application Structure Diagram", page 156.](#)
- an *application system technical architecture* used to represent the technical architecture chosen for the deployment of each of the applications that support the application system.
 For more details, see ["Creating an application system structure diagram", page 88.](#)

Creating an Application System

To create an *application system*:

1. From the **Application Architecture** navigation pane, select **Functional Architecture > Application Systems**.
The list of application systems appears in the edit area.
2. Select **My Application Systems**.
3. Click **New**.
The **Creation of Application System** dialog box appears.
4. Enter the **Name** of your application system and click **OK**.
The new application system appears in the list.

Application System Properties

The **Characteristics** properties page for an application system provides access to:

- its **Owner**, by default during creation of the application system, the current library.
- its **Name**,
- the text of its **Description**.

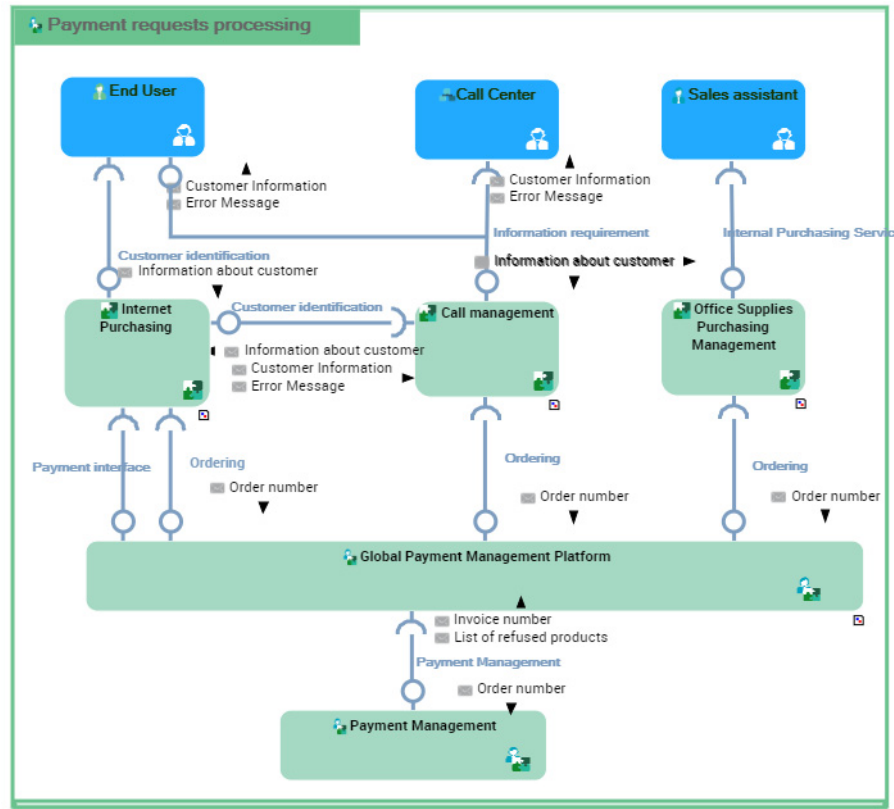
With **HOPEX IT Architecture** an application system is described by other property pages. See ["HOPEX IT Architecture properties pages content", page 47.](#)

Creating an application system structure diagram

This diagram describes the internal structure of an application system:

- services offered or required,
- the application components and their interactions; these are application systems, applications and micro-services,
- the end users interacting with the application components.

The following diagram describes the application system corresponding to purchasing requests processing.



The following diagram describes the application system corresponding to purchasing requests processing.

To create an application system structure diagram:


- 1 Right-click the application system and select **New > Application System Structure Diagram**.

Adding an application system to an application system structure diagram

To describe an application system that implements another application system, you can add an **application system** of the application system structure diagram.


For example, the purchasing requests processing system uses the "Purchasing Management Platform" and "Payment Management" application system services.

To add an **Application System**:




1. In the objects toolbar of the application system structure diagram, click  **Application System**.
2. Click in the frame of the described application system.
An addition window prompts you to choose the **application system** implemented (for example "Payment management").
3. Select an application system.
4. Click **OK**.
The application system appears in the diagram.

Adding an end user to an application system structure diagram

To specify that an application system, such as purchasing request processing, is activated by internal or external org-units, you will add an associated **end user**.

 *The end user represents an organizational unit interacting at the boundaries of an application system or a logical application system.*

To add an **end user**:

1. In the application system structure diagram insert toolbar, click  **End User** and click in the frame of the diagram.
An addition window prompts you to choose the **Object Type** that you wish to use: **Org-Unit** or **Position type**.
2. For example, select the **Org-unit** object type.
 *An org-unit represents a person or a group of persons that intervenes in the enterprise business processes or information system. An org-unit can be internal or external to the enterprise. An internal org-unit is an organizational element of enterprise structure such as a management, department, or job function. It is defined at a level depending on the degree of detail to be provided on the organization (see org-unit type). Example: financial management, sales management, marketing department, account manager. An external org-unit is an external entity that exchanges flows with the enterprise. Example: customer, supplier, government office.*
 *A position type represents a status assigned to an individual or a group of individuals with the aim of defining an organization or a hierarchy.*
3. Select the org-unit that interests you and click **OK**.
The actor appears in the diagram.

Describing an Application System Environment with HOPEX IT Architecture



An application system environment allows presenting the other application systems, applications or micro-services with which this application system can interact.

Accessing the list of application system environments

To access the list of application system environments from the **Application Architecture** navigation pane:

1. Select **Functional Architecture > Application System Environment**. The list of application system environments appears in the edit area.

Creating an application system environment

To create an *application system environment*:

1. From the **Application Architecture** navigation pane, select **Functional Architecture > Application System Environments**. The lists of application system environments appear in the edit area.
2. Click **New**. The **Creation of Application System Environment** window opens.
3. Enter the **Name** of your application system environment and click **OK**. The new application system environment appears in the list.

Application system environment properties

The **Characteristics** properties page for an application system environment provides access to:

- its **Owner**, by default during creation of an application system environment, the current library.
- its **Name**,
- the text of its **Description**.

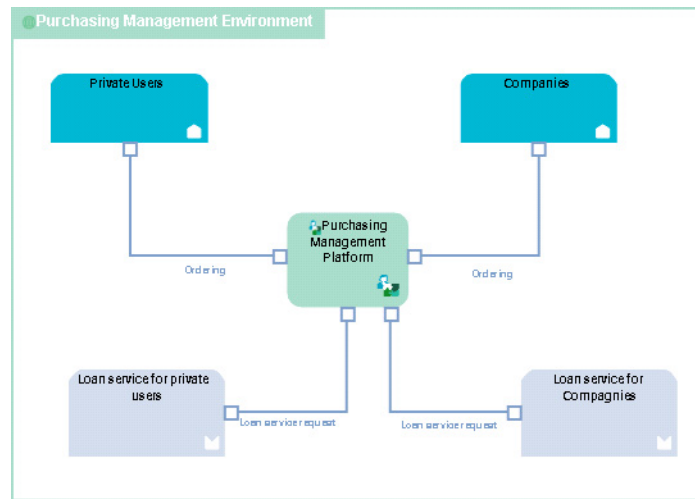
With **HOPEX IT Architecture** an application system environment is described by other property pages. See ["HOPEX IT Architecture properties pages content", page 47](#).

Describing an application system environment diagram

An application system environment is described by an application system environment diagram that describes the interactions between the internal application systems, its users and the partner application systems.



For more details on use of a structure diagram, see ["Creating a structure diagram", page 156](#)



Environment diagram for the "Purchasing Requests Processing" application system.

Purchase requests are formulated by private users or by companies in different contractual conditions.

The "Purchasing Request Processing" application system offers a loan service to its clients within the context of payment management.

The elements of an application system environment diagram are:

- the main **application system** principal described by the environment.
 An application system is an assembly of other application systems, applications and end users interacting with application components to implement one or several functions.
- partner application systems** that represent the other application system with which the main application system described by the environment interacts.

In this example, this concerns two loan services offered to individuals and companies.

A partner application system is an application system external to the environment of the described application service. The partner application system can be a service supplier or a service consumer with respect to application system users.

- The categories of users of services provided by the environment are represented either by an **Org-Unit** or by a **Position Type**.
 An org-unit represents a person or a group of persons that intervenes in the enterprise business processes or information system. An org-unit can be internal or external to the enterprise. An internal org-unit is an organizational element of enterprise structure such as a management, department, or job function. It is defined at a level depending on the degree of detail to be provided on the organization (see org-unit type). Example: financial management, sales management, marketing department, account manager. An external org-unit is an external entity that exchanges flows with the enterprise.

Example: customer, supplier, government office.



A position type represents a status assigned to an individual or a group of individuals with the aim of defining an organization or a hierarchy.

This concerns two user categories: individuals and companies.

- **interactions** between components



An interaction represents a contract established in a specific context between autonomous entities that are internal or external to an enterprise. These entities can be enterprise org-units, applications, activities or processes, as well as external org-units. The content of this contract is described by an exchange contract.

Describing an IT Service with HOPEX IT Architecture



An IT service is a software component of an application, that can't be deployed alone and that realizes a sub-set of the functionalities of this application either for end users of this application or inside the application (or another application). This includes batch programs.

Accessing the list of IT services

To access the list of IT Services from the **Business Architecture** navigation pane:

- 1 Select **Functional Architecture > IT Services**.

The list of IT services appears in the edit area.

IT Service properties

The complete description of an IT Service is accessed from its properties pages.

The **Characteristics** properties page for an IT Service provides access to:

- its **Owner**, by default during creation of the IT service, the current library.
- its **Name**,
- the text of its **Description**.
- the **Technologies** section provides access to the list of **Software Technologies** used by the IT Service.




A software technology is a basic component necessary for operation of business applications. Software technologies include all basic software such as: application server, electronic mail server, software components for presentation, data entry, storage, business information sharing, operating systems, middleware, navigators, etc.



For more details on software technologies, see ["Describing a Software Technology", page 105](#).

With **HOPEX IT Architecture** an IT service is described by other property pages. See ["HOPEX IT Architecture properties pages content", page 47](#).

Describing a Micro-Service with HOPEX IT Architecture

 A micro-service is a software component that can be deployed autonomously, but which does not directly provide an end user service. It can interact with other application services, applications or application systems. This is a deployable software component that uses software technologies. For example: an authentication service, a PDF file printing service.

Accessing the list of micro-services

To access the list of micro-services from the **Application Architecture** navigation pane:


- 1 Select **Functional Architecture > Micro-Services**.
The list of micro-services appears in the edit area.


Micro-Service properties with HOPEX IT Architecture

The complete description of a micro-service is accessed from its properties pages.

The **Characteristics** properties page for a micro-service provides access to:


- its **Owner**, by default, during creation of the micro service, the current library.
- its **Name**,
- the text of its **Description**.
- the **Technologies** section provides access to the list of *software technologies* used by the micro-services.

 A software technology is a basic component necessary for operation of business applications. Software technologies include all basic software such as: application server, electronic mail server, software components for presentation, data entry, storage, business information sharing, operating systems, middleware, navigators, etc.

 For more details on software technologies, see "[Describing a Software Technology](#)", page 105.

With **HOPEX IT Architecture** a micro-service is described by other property pages. See "[HOPEX IT Architecture properties pages content](#)", page 47.

Using the IT Service and Micro-Service Structure Diagram

 For more details on use of a structure diagram, see "[Creating a structure diagram](#)", page 156

With **HOPEX IT Architecture**, the components of an IT Service can be described by an IT Service structure diagram.

In the same way, the components of a micro-service can be described by a micro-service structure diagram.

These two diagrams include:

- IT services,
- micro services,
- Physical data stores; see ["Managing Data", page 96](#).
- access, request and service points; ["Describing Service and Request Points", page 169](#).
- *interactions* between the components



An interaction represents a contract established in a specific context between autonomous entities that are internal or external to an enterprise. These entities can be enterprise org-units, applications, activities or processes, as well as external org-units. The content of this contract is described by an exchange contract.

Describing Application Processes



A system process is the executable representation of a process. the events of the workflow, the tasks to be carried out during the processing, the algorithmic elements used to specify the way in which the tasks follow each other, the information flows exchanged with the participants.

Accessing the list of application processes

To access the list of application processes from the **Application Architecture** navigation pane:

- 】 Select **Functional Architecture > Application Processes**.
The list of application processes appears in the edit area.

Creating a system process diagram

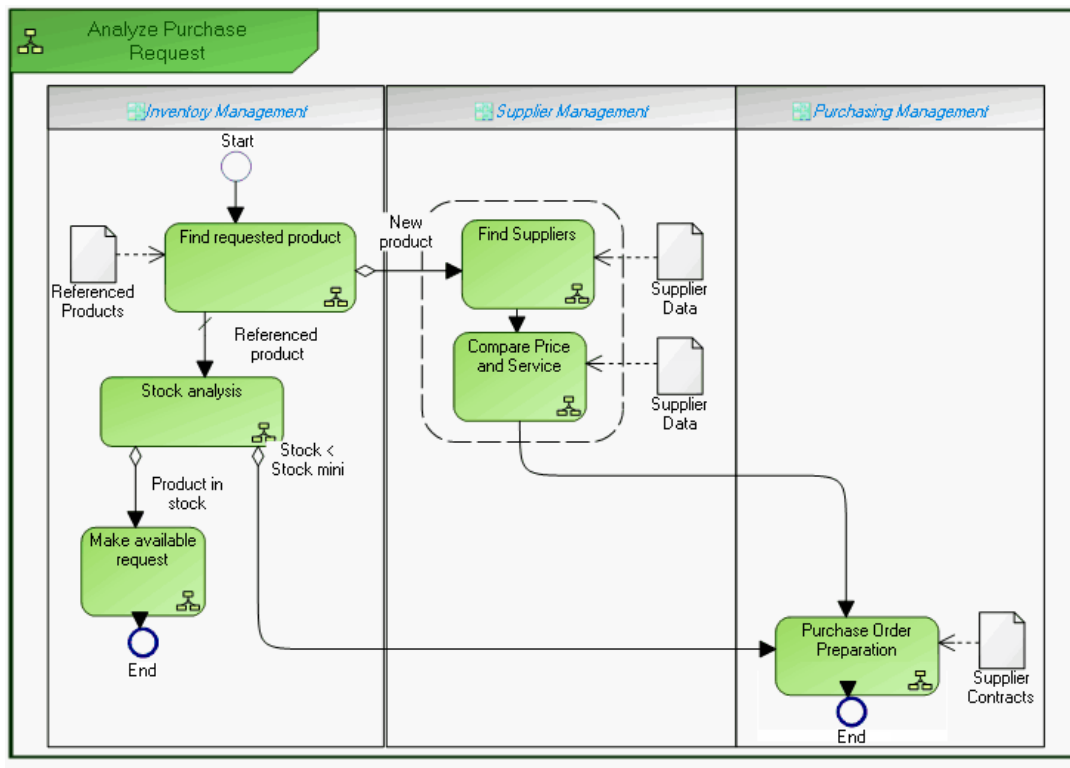
To create an application process diagram:

- 】 Right-click the application process that interests you and select **New > Application Process Diagram**.
The diagram opens in the edit area. The frame of the application process described appears in the diagram.

System Process example

The diagram below represents purchase request processing.

- A product search is carried out from the referenced products repository.
- If the product is new, search for a supplier and comparative study of prices is carried out. An order is then sent and the process ends.
- If the product is referenced, stock is analyzed.
- If stock is sufficient, a "Make available" request is activated and the process ends.
- If stock is less than minimum stock, an order is sent to the supplier and the process ends.



"Purchasing request" application process

➤ For more details on the use of functional processes, see the **HOPEX Business Process Analysis guide, chapter "System Processes", page 79.**

MANAGING DATA

Data stores are used in architecture diagrams to represent data that must be stored to be share between components.



A data store provides a mechanism to update or consult data that will persist beyond the scope of the current process. It enables storage of input message flows, and their retransmission via one or several output message flows.

A *data store* references an *data domain*.



A data area represents a restricted data structure dedicated to the description of a software Data Store. It is made of classes and/or data views and can be described in a Data Area Diagram.

Using Data Stores

Introduction to data store concept



A data area represents a restricted data structure dedicated to the description of a software Data Store. It is made of classes and/or data views and can be described in a Data Area Diagram.



*For more information on data areas, see chapter "Logical and application data areas" in the **HOPEX Information Architecture** guide.*

Accessing to data areas with HOPEX IT Architecture

To access the list of data areas from the **Application Architecture** navigation pane:

- 1. Select **Data management**.
Two tiles enable you to access to the **Logical data areas** and to the **Application data areas**.


Creating a data area

To create an application data area instance, for example:

To create an *application system environment*:


1. From the **Application Architecture** navigation pane, select **Data management > Application data areas**.
The lists of application data areas appear in the edit area.
2. Click **New**.
The new application data area appears in the list.
3. Modify the **Name** of your application data area.

Using Data Stores

 A data store provides a mechanism to update or consult data that will persist beyond the scope of the current process. It enables storage of input message flows, and their retransmission via one or several output message flows.


Introduction to the data store concept

A data store references an **data domain**.


 A data area represents a restricted data structure dedicated to the description of a software Data Store. It is made of classes and/or data views and can be described in a Data Area Diagram.

 For more information on data area, see chapter "Logical and application data areas" in the **HOPEX Information Architecture** guide.


If you describe a logical application system, only **logical data stores** can be used.


 A logical data store represents the use of data via application systems without considering how their access will be concretely implemented.


If you describe an application system, only **physical data stores** can be used.

 A physical data store represents the implementation of a logical data store.


If you describe scenario sequence or a scenario of flow, only **application data stores** can be used.


 An application data store materializes the usage of data in the context of a software component (for instance an application). An application data store provides a mechanism to retrieve or update information stored outside of the current software component.

 The Scenario of flows diagrams that describe the flows exchanged in different use scenarios of the object described.

 The scenario sequence of flow diagrams that describe the chronology of the flows exchanged in different use scenarios of the object described.

Last but not least, you can also distinguish data stores local to a system from external data stores that are positioned on the border of diagrams.

 A local data store represents a data store used only inside the system described.

 An external data store represents a data store used inside and outside of the system described.

Usage contexts

The table below presents the list of diagrams that use the different types of data stores.

| Data store type | Diagrams |
|-------------------------|---|
| Logical data store | Logical application system structure diagrams |
| Physical data store | Structure diagrams <ul style="list-style-type: none"> - of application, - of application system, - IT service, - of micro-service. |
| Application data stores | Scenario sequence diagrams <ul style="list-style-type: none"> - of application, - of application system, - IT service, - of micro-service. Scenario of flows diagrams <ul style="list-style-type: none"> - of application, - of application system, - IT service, - of micro-service. |

Creating a local data store




A local data store represents a data store used only inside the system described.

To create, for example, a **local physical data store** from an application system structure diagram:

1. Open the diagram that interests you.
2. In the diagram objects toolbar, click **Local physical Data Store**.
3. Click in the frame of the described application system.

A creation dialog box prompts you to select an existing **Data Area**.

 *The data area is the structure that will concretely support the data store. A physical or an application data store is supported by an **application data area**. A logical data store is supported by a **logical data area**.*

 *For more information on data areas, see chapter "Logical and application data areas" in the **HOPEX Information Architecture** guide.*

4. Select an existing **Data area**.
5. Click **OK**.

The local physical data store appears in the diagram with the name of the data area selected.

Creating an external data store



An external data store represents a data store used inside and outside of the system described.

To create, for example, an external physical data store from an application system structure diagram:

1. Open the diagram that interests you.
2. In the diagram objects toolbar, click **External physical Data Store**.

3. Click at the edge of the frame of the described application system.
A creation dialog box prompts you to select an existing **Data Area**.
 - ☛ *The data area is the structure that will concretely support the data store. A physical or an application data store is supported by an **application data area**. A logical data store is supported by a **logical data area**.*
 - ☛ *For more information on data areas, see chapter "Logical and application data areas" in the **HOPEX Information Architecture** guide.*
4. Select an existing **Data area**.
5. Click **OK**.
The local physical data store appears in the diagram with the name of the data area selected.

Describing access to a data store

To create a read access to the data store:

1. In the diagram insert toolbar, click **Link**.
2. Draw a link between the data store and the entity that reads the data (component or application system use).
A **Read-only access to data storage** is automatically created with the link from the data store to the entity.
 - ☛ *To create a link with write access, you must draw a link between this entity and the data store to which it has write access. A **Write access to data storage** is then automatically created.*

DESCRIBING APPLICATION FLOWS

The Global Application Flow Map is the point of entry for modeling which is used to represent the main exchanges between the application systems and the applications.

Creating a Global Application Flow Map

An application flow global map is used to represent all the flows exchanged between the elements of the application architecture. The elements represented are:

- application systems,
- applications,
- architecture users.

The interactions offered between these elements are application flows that carry a content.

Accessing the list of global application flow maps

To access the list of global application flow maps using the **Application Architecture** navigation pane:

- 】 Select **Functionality Architecture > Global Application Flow Map**.
The list of global maps for application flows appears in the edit area.

The properties of a global application flow map

The complete description of a global application flow map is accessed from its properties pages.

The **Characteristics** properties page of the global application flow map provides access to:

- its **Owner**, by default, on creation of the global application flow map, the current library.
- its **Name**,
- the text of its **Description**.

The **Scenario** property page of a global application flow map provides access to the list of its components: application systems, applications, participants and flows.

Using a Scenario of Application System Flows

☞ For more details on the use of a scenario of flows, see "[Describing a scenario of flows](#)", page 158.

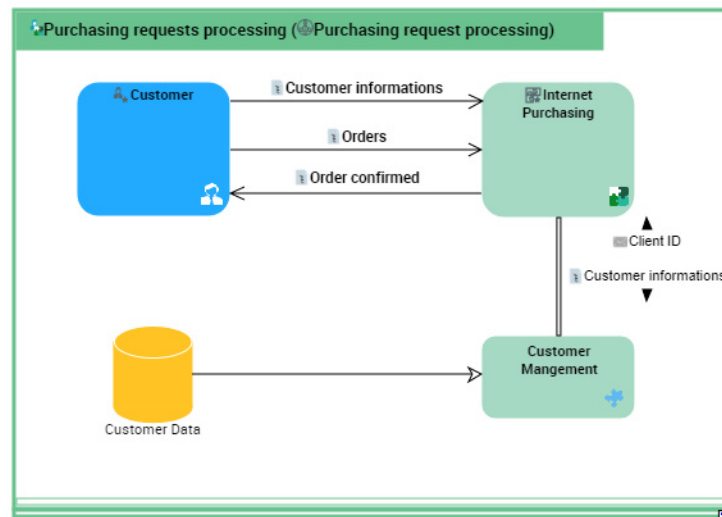
A scenario of application system flows represents the flows exchanged between certain elements of the application system in a given context. The elements represented are:

- application systems,
- applications,
- micro services,
- organization org-units,
- stores of internal or external application data,
- input or output application ports.

The interactions offered between these elements:

- application flows that carry a content,
- application flow channels that group a number of application flows on a single link,
- application data channels that represent the interactions between the application data stores.

The scenario of application system flows below describes the interactions between a client and the eCommerce application.




Example of scenario of application system flows for "Purchasing Requests Processing".

To create a scenario of application system flows:

1. Right-click the application system and select **New > Scenario of Application System Flows**.

Adding a scenario of application system flows

 An application is a software component that can be deployed and provides users with a set of functionalities.

To add an **application**:

1. In the scenario of application system flows object toolbar, click **Application**.

2. Click in the frame of the described application system.
An addition dialog box prompts you to choose the **application** that you want to use (for example "eCommerce purchase").
3. Select the application and click **OK**.
The application appears in the diagram.

In the same way you can add:

- an application system



An application system is an assembly of other application systems, applications and end users interacting with application components to implement one or several functions.

- a micro-service.



A micro-service is a software component that can be deployed autonomously, but which does not directly provide an end user service. It can interact with other application services, applications or application systems. This is a deployable software component that uses software technologies. For example: an authentication service, a PDF file printing service.

Adding an org-unit to the Scenario of Application System Flows

An org-unit is represented by an **Org-Unit** or by a **Position type**.




An org-unit represents a person or a group of persons that intervenes in the enterprise business processes or information system. An org-unit can be internal or external to the enterprise. An internal org-unit is an organizational element of enterprise structure such as a management, department, or job function. It is defined at a level depending on the degree of detail to be provided on the organization (see org-unit type). Example: financial management, sales management, marketing department, account manager. An external org-unit is an external entity that exchanges flows with the enterprise. Example: customer, supplier, government office.




A position type represents a status assigned to an individual or a group of individuals with the aim of defining an organization or a hierarchy.

To add an organization unit:

1. In the Scenario of Application System Flows object toolbar, click **Operator Participant**.
2. Click in the frame of the described application system.
An addition window prompts you to choose the **Object Type** that you wish to use:
3. For example, select the **Org-unit** object type.
4. Select the org-unit that interests you and click **OK**.
The actor appears in the diagram.

 To create a new org-unit, enter his name and click **OK**.

Using a Scenario of IT Service Flows and Micro-Service Flow

 For more details on use of a scenario of flows, see ["Describing a scenario of flows", page 158](#)

A Scenario of IT Service Flows or micro-service flow represents the flows exchanged between certain elements of the IT Service, or micro-service, in a given context. The elements represented are:

- IT services,
- micro services,
- stores of local or external application data,
- input or output application ports.

The interactions offered between these elements:

- application flows that carry a content,
- application flow channels that group a number of application flows on a single link,
- application data channels that represent the interactions between the application data stores.

Describing a Scenario of Application System Environment

A scenario of application system environment represents the flows exchanged between the components of the application system environment.

☛ For more details on use of a scenario of flows, see ["Describing a scenario of flows", page 158](#)

The elements of a scenario of application system environment are:

- the main **application system** principal described by the environment.



An application system is an assembly of other application systems, applications and end users interacting with application components to implement one or several functions.

- **partner application systems** that represent the other application system with which the main application system described by the environment interacts.



A partner application system is an application system external to the environment of the described application service. The partner application system can be a service supplier or a service consumer with respect to application system users.

- **End User Participants** that represent the categories of users of application system provided by the environment.
- The categories of users of services provided by the environment are represented either by an **Org-Unit** or by a **Position Type**.



An org-unit represents a person or a group of persons that intervenes in the enterprise business processes or information system. An org-unit can be internal or external to the enterprise. An internal org-unit is an organizational element of enterprise structure such as a management, department, or job function. It is defined at a level depending on the degree of detail to be provided on the organization (see org-unit type). Example: financial management, sales management, marketing department, account manager. An external

org-unit is an external entity that exchanges flows with the enterprise. Example: customer, supplier, government office.



A position type represents a status assigned to an individual or a group of individuals with the aim of defining an organization or a hierarchy.

- **interactions** between components




An interaction represents a contract established in a specific context between autonomous entities that are internal or external to an enterprise. These entities can be enterprise org-units, applications, activities or processes, as well as external org-units. The content of this contract is described by an exchange contract.

DESCRIBING TECHNICAL ARCHITECTURE

A technical architecture allows you to represent the elements that must be deployed to implement an application: *Application Technical Areas*, *Technical Data Areas* as well as *Technical Communication Lines* used for data exchange.

This description is based on *Software Technologies* and *Software technology Stacks*.

Describing a Software Technology

 A software technology is a basic component necessary for operation of business applications. Software technologies include all basic software such as: application server, electronic mail server, software components for presentation, data entry, storage, business information sharing, operating systems, middleware, navigators, etc.

Accessing the list of software technologies

To access the list of software technologies from the **Application Architecture** navigation pane:

- Select **Technical Architecture > Technologies**.
The list of software technologies appears in the edit area.

The properties of a technology

The complete description of a technology is accessed from its properties pages.


The **Characteristics** property page of a software technology provides access to:


- its **Owner**, by default during creation of the technologie, the current library.
- its **Name**,
- the text of its **Description**.
- its **Owned Realizations** which represent the list of functionalities, or logical applications, covered by this software technology.

A software technology is described by the **Use** which is used to obtain the list of technical architectures that use the software technology.

➡ For more details on technical architectures, see [and "Creating an Application Technical Architecture", page 106.](#)

Describing Software Technology Stacks

 A software technology stack is a set of software technologies.

 A software technology is a basic component necessary for operation of business applications. Software technologies include all basic software such as: application server, electronic mail server, software

components for presentation, data entry, storage, business information sharing, operating systems, middleware, navigators, etc.

Accessing the list of technology stacks

To access the list of software technology stacks from the **Application Architecture** navigation pane:

- 1 Select **Technical Architecture > Software Technology Stack**.


The list of software technology stacks appears in the edit area.

Properties of a technology stack

The complete description of a software technology stack can be accessed from its property pages.

The **Characteristics** property page of a software technology stack provides access to:


- its **Owner**, by default, on creation of the software technology stack, the current library.
- its **Name**,
- the text of its **Description**.
- its **Owned Realizations** which represent the list of functionalities covered by this technology stack.

 For more details on functionalities, see ["Describing a Functionality Map with HOPEX IT Architecture"](#), page 67.

Creating an Application Technical Architecture

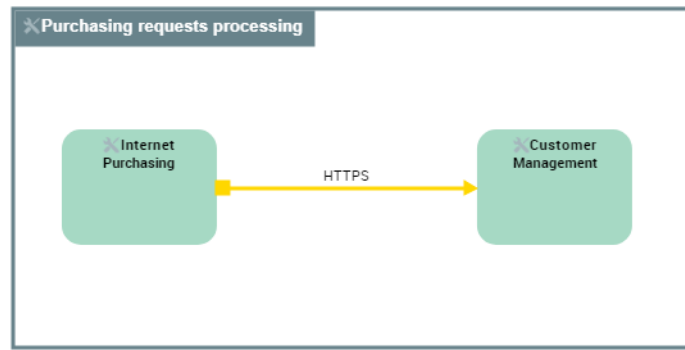


An application technical architecture describes one of the configurations possible for application deployment. It describes how the different technical areas of the application are connected to each other and the technologies and the communication protocols that they use. An application can have a number of possible technical architectures (E.g.: autonomous installation, horizontal or vertical deployment, etc.)

 For more details on the use of a technical architecture, see ["Describing a Computing Device"](#), page 118.

The application technical architecture below presents the application technical areas for eCommerce and customer management. The technical communication line between these

two components is based on the https communication protocol.



Example of the application technical architecture for "Purchasing Request Processing".

An application technical architecture is described by an application technical architecture diagram composed of the following elements:

- **Application Technical Areas,**
 An application technical area represents an organizational element of an application according to technical criteria. For example, this can be the user interface or a process. Each application technical area is associated with one or more technologies. The deployment of several application technical areas is necessary for the application to be operation.
- **Technical Data Areas,**
 A data technical area represents an organizational element of an application used to access the data necessary for the operation of this application. Each application technical area is associated with one or more technologies (E.g.: Oracle 12, SQL Server 2012, etc.). A data technical area can allow access to one or more data stores.
- **Technical Communication Lines.**
 A technical communication line represents a technical connection between architectures or application technical areas through client and server ports. Client technical port of an architecture or a technical area requires opening the communication line to server technical port of the other area or technical architecture.

Accessing the application technical architectures

To access the list of application technical architectures from the **Application Architecture** navigation pane:

- › Select **Technical Architecture > Technical Architectures**.
The list of technical architectures appears in the edit area.

Properties of an application technical architecture

The complete description of an application technical architecture can be accessed from its property pages.

The **Characteristics** property page of an application technical architecture provides access to:

- Its **Owner**, by default the application specified when it was created.
- its **Name**,
- the text of its **Description**.

With **HOPEX IT Architecture** an application technical architecture is described by other property pages. See ["HOPEX IT Architecture properties pages content", page 47](#).



MODELING IT INFRASTRUCTURES



Functionalities proposed by **HOPEX IT Architecture** for modeling complex infrastructures enable representation of equipment, IT and organizational resources required for system deployment and operation: interactions between components, communication means supporting these interactions, and services offered and used by the modeled architecture.

The following points are covered here:

- ✓ ["Describing a Resource Architecture", page 112;](#)
- ✓ ["Describing IT Components", page 117;](#)
- ✓ ["Describing communications in a Technical Infrastructure", page 122.](#)

DESCRIBING A RESOURCE ARCHITECTURE

A *resource architecture* comprises equipment, IT and organizational resources required for operation of a complex infrastructure (system).

Communications between these components are represented by interactions and the equipment means supporting these interactions are the communication channels.



A resource architecture is the combination of physical and organizational assets configured to supply a capability.

Services offered by the system to its users are represented by service points. Service points are physically supported by communication ports that enable access to communication means of the system.

Creating a Resource Architecture

To create a resource architecture:

1. From the **Infrastructure** navigation pane, select **Infrastructure > Resource Architecture**.
The list of resource architectures appears.
2. Click **New**.
The **Creation of a Resource Architecture** window opens.
3. Enter the **Name** of your architecture as well as its **Owner** and click **OK**.
The new resource architecture appears in the list.

Managing a Resource Architecture Assembly Diagram

Creating a Resource Architecture Assembly Diagram:

To create a Resource Architecture Assembly Diagram:


1. Right-click the resource architecture and select **New > Resource Architecture Assembly Diagram**.
The diagram opens in the edit area.

Adding a Resource Architecture


To describe that a resource architecture, such as a call center, implements another resource architecture, such as a customer management service for example, you will add the resource architecture used in the user resource architecture diagram.


To add a resource architecture to a Resource Architecture Assembly Diagram:

1. In the objects toolbar of the Resource Architecture Assembly Diagram, click **Resource Architecture**.

2. Click in the frame of the Resource Architecture Assembly Diagram.
An addition dialog box prompts you to select a resource architecture. You can select an existing resource architecture or create a new one.
 *To create a new resource architecture, simply enter its name.*
3. Click **OK**.

Adding an Org-Unit or a Position Type


 *An org-unit represents a person or a group of persons that intervenes in the enterprise business processes or information system. An org-unit can be internal or external to the enterprise. An internal org-unit is an organizational element of enterprise structure such as a management, department, or job function. It is defined at a level depending on the degree of detail to be provided on the organization (see org-unit type). Example: financial management, sales management, marketing department, account manager. An external org-unit is an external entity that exchanges flows with the enterprise. Example: customer, supplier, government office.*


 *A position type represents a status assigned to an individual or a group of individuals with the aim of defining an organization or a hierarchy.*

To describe that a Resource Architecture such as a call center uses operators to take calls and handle requests, you will create a **Position Type** component.

To add a Position Type to a Resource Architecture Assembly Diagram:


1. In the objects toolbar of the Resource Architecture Assembly Diagram, click **Position Type**.
2. Click in the frame of the resource architecture described.
An addition window prompts you to choose the **Position Type** that you wish to use:
3. Select the Position Type concerned and click **OK**.
The Position Type appears in the diagram.

 *To create a Position Type, simply enter its name.*

 *In the same way, you can add an org-unit in the Resource Architecture Assembly Diagram.*


Describing Resource Architecture Technical Resources

The network resources group both IT resources, such as servers, networks and devices, and non IT resources such as vehicles for example.

 *For more details on hardware resources, see "Describing IT Components", page 117.*

Adding an IT infrastructure

To describe that a resource architecture is based on IT resources such as a communication network, workstations housing applications, you will add **IT Infrastructure** type components to the Resource Architecture Assembly Diagram.

 *An IT infrastructure is made up of different IT hardware components such as: IT technical devices, computers or IT networks.*

To create an **IT Infrastructure**:

1. In the diagram objects toolbar, click **IT Infrastructure**.

2. Click in the diagram frame.
An addition dialog box prompts you to select the IT infrastructure to be deployed.
3. Select the IT infrastructure that interests you and click **OK**.
The IT infrastructure appears in the diagram.

☛ *To create an IT infrastructure type, simply enter its name.*

☛ *In the same way, you can add another hardware resource in the Resource Architecture Assembly Diagram. For more details on hardware resources, see ["Describing IT Components"](#), page 117.*

Channels and communication ports

In a resource architecture, communication channels support the transfer of information from one hardware asset to another. For more details on creation of these channels and the associated communication protocols, see ["Communication channels"](#), page 123.

Communication ports enable connection of resource architecture physical assets with external equipment elements.

Describing the Services in a Resource Architecture Assembly Diagram

Describing Resource Architecture Services and Requests

A resource architecture is created to assure one or several services.

☛ *For more details, see ["Service points"](#), page 122 and ["Request points"](#), page 122.*

Describing Interaction in a Resource Architecture Assembly Diagram

In a resource architecture assembly diagram, **Interactions** enable representation of exchanges between organizational entities.

📖 *An interaction represents a contract established in a specific context between autonomous entities that are internal or external to an enterprise. These entities can be enterprise org-units, applications, activities or processes, as well as external org-units. The content of this contract is described by an exchange contract.*

Exchange terms are defined by an **Exchange contract** assigned to the interaction.

📖 *An exchange contract is a model of a contract between organizational entities. This contract is described by exchanges between an initiator role and one or several contributor roles.*

You can define interactions between:

- Two components of resource architecture type to represent exchanges between these entities,
- A component of resource architecture type and an IT infrastructure to represent the terms of use of the equipment resource by the

organizational resource. For example, you can represent that operator hardware use is arranged by booking.

- two components of IT infrastructure type to represent the terms of use of one IT resource by another in the context of the modeled resource architecture.
- a service point and one or more resource architecture type components to represent implementation of the service within the resource architecture,
- A component of architecture use type and a request point to represent that the entity calls a resource of an external organization.

For more details on interaction management terms, see ["Describing Communications at Equipment Level", page 123](#).

Describing a Resource Architecture Environment



A business architecture environment represents the relationships of a business functional area with its partners.

Creating a resource architecture environment

To create a resource architecture environment:

1. From the **Infrastructure** navigation pane, select **Infrastructure > Resource Architecture Environments**.
The list of resource architecture environments appears.
2. Click **New**.
The **Creation of a Resource Architecture Environment** window opens.
3. Enter the **Name** of your environment as well as its **Owner** and click **OK**.
The new architecture environment appears in the list.

To create a resource architecture environment diagram

To create a resource architecture environment diagram:

1. Right-click the resource architecture environment and select **New > Resource Architecture Environment Diagram**.
The diagram opens in the edit area.

Adding a Resource Architecture




A resource architecture is the combination of physical and organizational assets configured to supply a capability.

To add a resource architecture, such as a call center, to a resource architecture environment:

1. In the objects toolbar of the resource architecture environment, click **Resource Architecture**.

2. Click in the frame of the resource architecture environment described. An addition dialog box prompts you to choose the resource architecture implemented (for example, "eCommerce purchase"). You can select an existing resource architecture or create a new one.

 In the case where the resource architecture you want to use does not yet exist in the repository, simply enter its name.

3. Click **OK**.

Creating a Partner Resource Architecture

To describe that an external resource architecture, such as a processing center for purchase requests, is implemented in the call center environment, you will add a *partner resource architecture* component type in the environment diagram.



A partner resource architecture is the installation of an external resource architecture in another resource architecture or an environment.

To create a **Partner Resource Architecture**:

1. In the objects toolbar of the resource architecture environment diagram, click **Partner Resource** and select **Resource Architecture**.
2. Click in the frame of the resource architecture environment described. An addition dialog box prompts you to select a resource architecture to add. You can enter the name of a new resource architecture.
3. Click **OK**.

Creating a human asset


To describe that the resource architecture environment services are used by customers, for example, you will create a *position type* or an *org-unit*.



A position type represents a status assigned to an individual or a group of individuals with the aim of defining an organization or a hierarchy.



An org-unit represents a person or a group of persons that intervenes in the enterprise business processes or information system. An org-unit can be internal or external to the enterprise. An internal org-unit is an organizational element of enterprise structure such as a management, department, or job function. It is defined at a level depending on the degree of detail to be provided on the organization (see org-unit type). Example: financial management, sales management, marketing department, account manager. An external org-unit is an external entity that exchanges flows with the enterprise. Example: customer, supplier, government office.

 For more details on creating a human asset, see ["Adding an Org-Unit or a Position Type", page 113](#).

DESCRIBING IT COMPONENTS

Describing an IT infrastructure



An IT infrastructure is made up of different IT hardware components such as: IT technical devices, computers or IT networks.

You can describe the components of an *Infrastructure* in an infrastructure assembly diagram.

Creating an IT infrastructure

To create an IT infrastructure:

1. From the **Infrastructure** navigation pane, select **Infrastructure > IT Infrastructure**.
The list of IT infrastructures appears.
2. Click **New**.
The **Creation of IT Infrastructure** window appears.
3. Enter the **Name** of your infrastructure as well as its **Owner** and click **OK**.
The new IT infrastructure appears in the list.

Creating an Infrastructure Assembly Structure Diagram

To create an infrastructure assembly structure diagram:

1. Right-click the IT Infrastructure and select **New > Infrastructure Assembly Structure Diagram**.
The diagram opens in the edit area.

Using an Infrastructure Assembly Structure Diagram

In an infrastructure assembly structure diagram, you can insert:

You can insert in this diagram:

- *IT servers* and *IT devices*, see "Describing a Computing Device", page 118,



An IT Server is an IT component providing a service to users connected via an IT network. This IT component can house databases and run applications.



An IT device is a computer that provides a service directly to the end user. This computer can house databases and run applications. This is, for example, a workstation, a laptop or a smartphone.

- *IoT Devices*,



An IoT device is both a hardware device and a computing device which provides combined hardware and information services to the users using it directly. As a hardware device, it embeds sensors - e.g. accelerometer - which provide data to the embedded computing device. As a computing device, it can host data stores or run applications.

Examples: smartwatch with GPS tracker, on-line surveillance video camera with live IP video feed, connected weighting scale with weight history management

- **IT Peripheral Device,**



An IT Peripheral Device can host and run Software Technology. Conjointly with its hosted software, it provides technical services. This consists of, for example, Examples: Wifi Access Point, Firewall, router, switch, printer, Hard Drive.

- **IT Network Components,** see ["Describing an IT network", page 120,](#)



An IT network is set of IT equipment components (e.g.: routers, switches, firewalls) that allow remote communications between computing devices (e.g.: IT server). An IT network can be broken down into sub-networks.

- communication ports and channels,
- service and request points,

☛ For more details, see ["Service points", page 122](#) and ["Request points", page 122.](#)

- interactions.



An interaction represents a contract established in a specific context between autonomous entities that are internal or external to an enterprise. These entities can be enterprise org-units, applications, activities or processes, as well as external org-units. The content of this contract is described by an exchange contract.

☛ For more details on interactions, see ["Describing Communications at Equipment Level", page 123.](#)

The **Infrastructure description** report allows you to analyze the infrastructure components and the expected relationships between them to verify that each element described in the diagram hosts a software or hardware component.

☛ For more details, see ["Infrastructure Description Report", page 196.](#)

Describing a Computing Device

Accessing the list of computing devices

To access all the different types of computing devices:

1. From the **Infrastructure** navigation pane, select **Infrastructure > Computing Devices.**
2. Select **All Computing Devices** tab.

The list of all computing devices appears:

- **Servers,**
- **IT Devices,**



An IT device is a computer that provides a service directly to the end user. This computer can house databases and run applications. This is, for example, a workstation, a laptop or a smartphone.

- **IoT Devices.**

Creating a computing device

To create a **computing device**:





1. From the **Infrastructure** navigation pane, select **Infrastructure > Computing Devices**.
2. Select **All Computing Devices** tab.
The list of all computing devices appears.
3. Click **New**.
4. Select the computing device type that you want to create and click **Next**.
5. Enter the **Name** of your computing device as well as its **Owner** and click **OK**.
The new computing device appears in the list.

Creating a Computing Device Assembly Diagram

To create a computing device assembly diagram:

1. Right-click the computing device and select **New > Computing Device Assembly Diagram**.
The diagram opens in the edit area.

You can insert the following in a computer assembly diagram:

- *application technical area host*, see ["Creating an application technical area host in a computing device assembly diagram"](#), page 120,
 *An application technical area represents an organizational element of an application according to technical criteria. For example, this can be the user interface or a process. Each application technical area is associated with one or more technologies. The deployment of several application technical areas is necessary for the application to be operation.*
- *software technology host*,
 *A software technology is a basic component necessary for operation of business applications. Software technologies include all basic software such as: application server, electronic mail server, software components for presentation, data entry, storage, business information sharing, operating systems, middleware, navigators, etc.*
- *micro-service host*,
 *A micro-service is a software component that can be deployed autonomously, but which does not directly provide an end user service. It can interact with other application services, applications or application systems. This is a deployable software component that uses software technologies. For example: an authentication service, a PDF file printing service.*
- *data store host*,
 *A data store provides a mechanism to update or consult data that will persist beyond the scope of the current process. It enables storage of input message flows, and their retransmission via one or several output message flows.*
- communication ports and channels,
- service and request points,
- interactions.

Creating an application technical area host in a computing device assembly diagram

To describe that an computing device hosts applications such electronic mail, you will create an **Application Technical Area Host** and associate the required application with it.



An application technical area represents an organizational element of an application according to technical criteria. For example, this can be the user interface or a process. Each application technical area is associated with one or more technologies. The deployment of several application technical areas is necessary for the application to be operation.

To create an application technical area host:

1. In the objects toolbar of the computer assembly diagram, click **Application Technical Area Host**.
2. Click in the diagram frame.
An addition dialog box prompts you to select a housed **Application**.
3. Select for example the electronic mail application and click **Add**.
4. Then create the associated application technical area and click **OK**.

Communication channels and ports in a computer assembly diagram

Communication channels support transfer of information between equipment resources. For more details on creation of these channels and the associated communication protocols, see "[Communication channels](#)", page 123.

Communication ports enable connection of IT equipment resources with external equipment elements.

Describing an IT network



An IT network is set of IT equipment components (e.g.: routers, switches, firewalls) that allow remote communications between computing devices (e.g.: IT server). An IT network can be broken down into sub-networks.

Creating an IT network

To create an IT network:

1. From the **Infrastructure** navigation pane, select **Infrastructure > IT Networks**.
The list of IT networks appears.
2. Click **New**.
The **IT Network Creation** window appears.
3. Enter the **Name** of your network as well as its **Owner** and click **OK**.

Creating an IT network

An IT network is described by an infrastructure assembly structure diagram.

To create an infrastructure assembly structure diagram from an IT network:

- 1 Right-click the IT network and select **New > Infrastructure Assembly Structure Diagram**.

The diagram opens in the edit area.

☛ For more details on this type of diagram, see "[Creating an Infrastructure Assembly Structure Diagram](#)", page 117.


DESCRIBING COMMUNICATIONS IN A TECHNICAL INFRASTRUCTURE

In an infrastructure, communications are based on:

- service points, request points and interaction points for organizational communications,
- communication ports and channels for technical communications.

Describing services and requests

Service points

Services assured by resource architecture elements are represented by *service points* .



A service point is a point of exchange by which an agent offers a service to potential customers.

The service is requested according to precise terms defined by an *exchange contract* assigned to the service point.



An exchange contract is a model of a contract between organizational entities. This contract is described by exchanges between an initiator role and one or several contributor roles.

Resources activated to assure a service are linked to the service point by interactions. If activation of several resources is necessary, then several interactions must be created between the service point and the architecture resources.

To create a service point, see ["Creating a Service Point or a Request Point", page 171.](#)

Request points

A *request point*  enables representation of use of an external service.



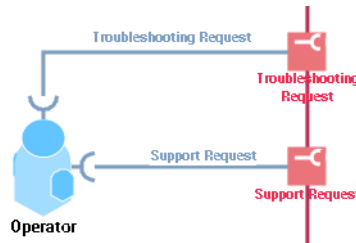
A request point is a point of exchange by which an agent requests a service from potential suppliers.

The service is requested according to precise terms defined by an *exchange contract* assigned to the request point.



An exchange contract is a model of a contract between organizational entities. This contract is described by exchanges between an initiator role and one or several contributor roles.

Resources that issue a request are linked to the request point by an interaction.



In the example, request points represent service requests made between call center operators and other organizations.

The request point creation procedure is identical to that for service points. For more details, see ["Creating a Service Point or a Request Point", page 171](#).

Describing Communications at Equipment Level

Communication ports

Communication Ports are physical points of communication that can be defined in technical infrastructures and resource architectures.

A communication port is a physical point of communication with a resource. It adheres to the specific communication protocol. A communication port implements service and requests points.

Communication Ports assure physical transfer of information exchanged on service points and request points.

Communication ports comply with specific "Communication Protocols". See ["Network communication protocols", page 124](#).

Communication channels

Communication Channels enable connection of equipment resources between themselves, with organizational resources or with communication points.

A communication enables physical connection between two equipment resources. It supports interactions that define communication between these resources. Communication channels connect resources with the exterior via communication ports.

Creating a communication channel

To create a communication channel:

1. In the resource architecture assembly diagram objects toolbar, click **Communication Channel** .
2. Draw a link between the two communication entities.
The channel appears directly in the diagram.

To define the communication protocol associated with the channel:

1. Open the **Supported Protocols** property page and click **Connect**.

2. In the query window that appears, select the communication protocol that interests you and click **Connect**.
The protocol name appears alongside the channel.

Network communication protocols

A *Communication Protocol* is supported by a communication channel.



A communication protocol is a set of standardized rules for transmission of information (voice, data, images) on a communication channel. The different layers of protocols can handle the detection and processing of errors, authentication of correspondents, management of routing.

For example, an HTTPS protocol is based on an HTTP protocol for transport, those protocols are based on TCP, which is itself based on Ethernet.

A user may wish to build a customized layer of communication protocols and assign these to communication ports and communication channels.



Communication protocols supported by a communication port must be compatible with the communication ports to which they are connected.



USING SERVICE CATALOGS



A service catalog describes the list of functionalities covered by a service as well as the technical or functional elements that implement these functionalities.



A service catalog contains a list of key service offers for which solutions are recommended.

The following points are covered here:

- ✓ ["Introducing service catalogs", page 128;](#)
- ✓ ["Populating a service catalog", page 131;](#)
- ✓ ["Service catalog reports", page 134.](#)

INTRODUCING SERVICE CATALOGS

The types of service catalogs

HOPEX IT Architecture offers the following service catalogs:

- *technical service catalogs,*



A business service catalog provides a centralized information source for the business services offered by the service provider organization. It contains a customer-oriented view of the services associated to business capabilities, how they are supposed to be used, the processes that they support as well as the expected service quality level. The business service catalog presents the list of functionalities mentioned as well as implementation recommendations.

- *information service catalogs,*



An information service catalog provides a centralized information source for the information services offered by the service provider organization. It contains a customer-oriented view of the information services used, how they are supposed to be used, the processes that they support as well as the expected service quality level. The information service catalog presents the list of functionalities mentioned as well as implementation recommendations.

- *technical service catalogs,*



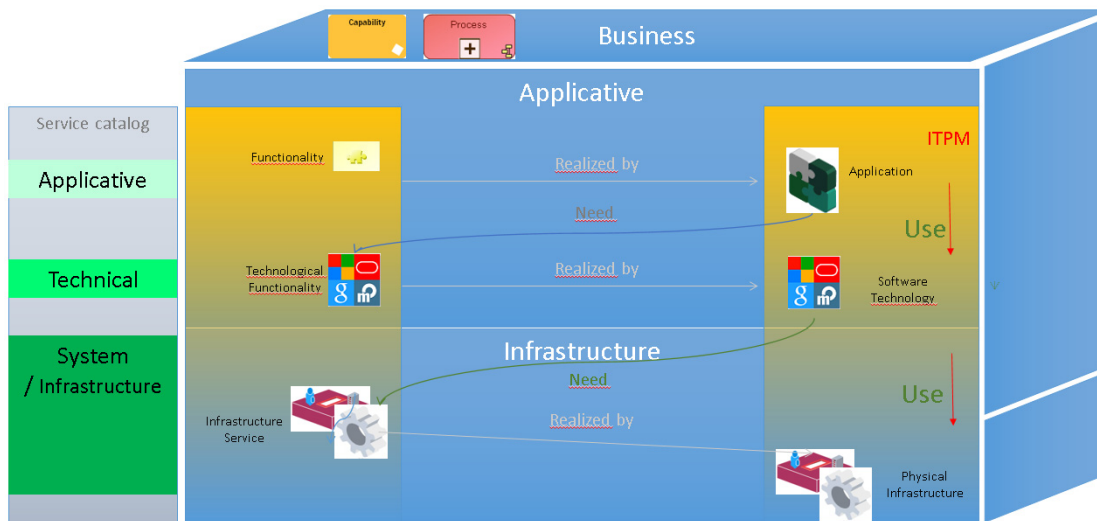
A technical service catalog provides a centralized information source for the technical services offered by the service provider organization. It contains a customer-oriented view of the technical services used, how they are supposed to be used, the processes that they support as well as the expected service quality level. The technical service catalog presents the list of IT functionalities mentioned as well as implementation recommendations.

- *hardware service catalogs.*



A hardware service catalog provides a centralized information source for the hardware services offered by the service provider organization. It contains a customer-oriented view of the hardware used, how they are supposed to be used, the processes that they support as well as the expected service quality level. The hardware service catalog presents the list of hardware functionalities mentioned as well as implementation recommendations.

The figure below presents the positioning of service catalogs offered by **HOPEX IT Architecture**.



Performing a functionality

In **HOPEX IT Architecture**, a service catalog is made up of service catalog item. For example, an *information service catalog* is made up of several *information service catalog items*.

An information service catalog provides a centralized information source for the information services offered by the service provider organization. It contains a customer-oriented view of the information services used, how they are supposed to be used, the processes that they support as well as the expected service quality level. The information service catalog presents the list of functionalities mentioned as well as implementation recommendations.

An information service catalog item defines which functionality is in the catalog and which application artifacts provide the functionality.

A *service catalog item* is connected to one or more *functionalities*.

A functionality is a service required by an org-unit in order to perform its work. This functionality is generally necessary within an activity in order to execute a specific operation. If it is a software functionality, it can be provided by an application.

With **HOPEX IT Architecture**, the implementation of a functionality is represented by a *realization*.





A realization describes the relationship between a logical entity and a physical entity that implements it. The physical entity gives the list of logical entities that it implements.

The table below draws up the summary of objects that implement the service catalogs according to their category.

| Type of service catalog | Type of functionality | Types of object that deliver the service |
|-------------------------|---------------------------|---|
| Business | Features | Business capability |
| Information | Features | All types of technical and functional objects that implement a functionality with HOPEX IT Architecture . For more details, see "Describing a Functionality Map with HOPEX IT Architecture" , page 67. |
| Technical | Technical functionalities | All types of technical objects implement a functionality with HOPEX IT Architecture : technical infrastructures, equipment and IT infrastructures, micro services, technical zones, all IT equipment (servers, networks, devices). |
| hardware | Hardware functionalities | Hardware and IoT Device. |

POPULATING A SERVICE CATALOG


The management principle of a service catalog is identical for all types of service catalogs. The types of service catalogs offered are:

- **technical service catalogs,**
 *A business service catalog provides a centralized information source for the business services offered by the service provider organization. It contains a customer-oriented view of the services associated to business capabilities, how they are supposed to be used, the processes that they support as well as the expected service quality level. The business service catalog presents the list of functionalities mentioned as well as implementation recommendations.*
- **information service catalogs,**
 *An information service catalog provides a centralized information source for the information services offered by the service provider organization. It contains a customer-oriented view of the information services used, how they are supposed to be used, the processes that they support as well as the expected service quality level. The information service catalog presents the list of functionalities mentioned as well as implementation recommendations.*
- **technical service catalogs,**
 *A technical service catalog provides a centralized information source for the technical services offered by the service provider organization. It contains a customer-oriented view of the technical services used, how they are supposed to be used, the processes that they support as well as the expected service quality level. The technical service catalog presents the list of IT functionalities mentioned as well as implementation recommendations.*
- **hardware service catalogs.**
 *A hardware service catalog provides a centralized information source for the hardware services offered by the service provider organization. It contains a customer-oriented view of the hardware used, how they are supposed to be used, the processes that they support as well as the expected service quality level. The hardware service catalog presents the list of hardware functionalities mentioned as well as implementation recommendations.*

This chapter is based on the example of an **information service catalog**.

Creating an information service catalog

To create an **information service catalog**:

1. From the **Catalogs** navigation pane, select **Information service catalog > Information service catalog**.
The lists of information service catalogs appear in the edit area.
2. Click **New**.
A new information service catalog appears in the edit area.
3. Enter the **Name** of your catalog and its **Owner**.
 *In the same way, you can create a **technical service catalog** or a **hardware service catalog**.*

Accessing the list of service catalogs

To access the list of *information service catalogs*

1. From the **Catalogs** navigation pane, select **Information service catalog > Information service catalog**.

The lists of information service catalogs appear in the edit area.

A service catalog is defined by all of its *information service catalog items*.



An information service catalog item defines which functionality is in the catalog and which application artifacts provide the functionality.

To access the list of *information service catalog items*:

1. From the **Catalogs** navigation pane, select **Information service catalog > Information service catalog items**.

2. In the **Information service catalog** field, select the catalog that interests you.

The tree of information service catalog item appears in the edit area.

Adding a service catalog item

The **Characteristics** property page of a service catalog provides access to:

- its **Owner**, by default, during creation of the logical application system, the current library.
- its **Name**,
- To access the list of service catalog items owned:

To add an *information service catalog item*:

1. Open the **Characteristics** properties page of an information service catalog.
2. In the **Information Service Catalog items** section, click **New**. In the window that opens, you can select an existing functionality or create a new one.
3. Select the functionality that interests you and click **OK**. The catalog element appears in the list with the name of the associated functionality.

Specifying the implementation of a service catalog item

To describe all the Service Catalog elements that implement a functionality, you will define a *Realization*.



A realization describes the relationship between a logical entity and a physical entity that implements it. The physical entity gives the list of logical entities that it implements.

To describe that a functionality is implemented by a Service Catalog element:

1. Open the **Characteristics** properties page of an information service catalog.
2. In the **Information Service Catalog items** section, click **New**. An add functionality dialog box appears:
3. Select the functionality that interests you.
4. Click **OK**. The functionality realization appears in the properties page.

SERVICE CATALOG REPORTS




With a report, you can determine, for a list of service catalogs, the list of functionalities covered, and for each of them, the implementation means.

Report example

The example below shows the functionalities covered by several service catalogs as well as elements that implement these functionalities.

1. Services coverage matrices

Retail Corp - IS Service Catalog

| | Description | proposed solution 1 |
|--|--|--|
|  Client & Customer Billing | Billing upon reservation must be available via mobile device |  Billing |
|  Consult Prices list | |  Catalog Management |

Retail Corp - Technical Service Catalog

| | Description | proposed solution 1 |
|---|-------------|--|
|  Application Server Functionalities | |  Retail Corp Central Infrastructure |

Report parameters

This consists of defining report input data.

| Parameters | Parameter type | Constraints |
|--------------------------|--|----------------------|
| List of service catalogs | All types of service catalogs: <ul style="list-style-type: none">- information,- technical,- hardware. | A mandatory catalog. |

Creating a service catalog report

To create a service catalog report:

1. From the **Catalogs** navigation pane, select **Reports**.
A new report opens in the edit area.
2. In the **Parameters** section, in the **Service Catalog** field, select the service catalogs that you want to present in your report.
3. Click **Refresh the Report**.
The new report appears.



MANAGING IT TRANSFORMATION



HOPEX IT Strategy is an option of the **HOPEX IT Architecture** product.

HOPEX IT Strategy is based on the tools of the **HOPEX** platform as well as the method embedded in the **HOPEX Business Architecture** solution to support the description, analysis and transformation projects of the IT system.

- ✓ [Overview of HOPEX IT Strategy;](#)
- ✓ [Managing IT Architecture Transformation With HOPEX IT Strategy.](#)

OVERVIEW OF HOPEX IT STRATEGY

Combined with the products of the **HOPEX** suite, **HOPEX IT Strategy** supports a methodology and the tools used to describe, analyze and plan your information system transformation.

HOPEX IT Strategy Profiles

In **HOPEX IT Strategy**, there are default user profiles with which specific rights and accesses are associated.

- **IT Strategist;**
- **Strategic Planning Functional Administrator.**

IT strategist

The IT strategist is the business user profile of the **HOPEX IT Strategy** solution.


The IT strategist is responsible for creating and structuring data relating to the IT system transformation.

If your license allows, and so that the users connected to this profile can integrate their work, the **IT strategist** can also access the objects and main functionalities of the **HOPEX IT Architecture** solution, via the **HOPEX IT Strategy** desktop.

 For more details on the **IT strategist** desktop, see [Presenting the IT Strategist desktop](#).

Strategic Planning Functional Administrator

The strategic planning functional administrator has extended rights on all managed objects. The strategic planning functional administrator is also in charge of the work organization of IT strategies.

 For more details on the functional administrator desktop for the solution, see [Presenting the functional administrator desktop](#).

Connecting to the solution

To connect to **HOPEX IT Strategy**, see HOPEX Common Features, "HOPEX Web Front-End Desktop".

HOPEX IT Strategy Desktop

The menus and commands available in **HOPEX IT Strategy** depend on the profile with which you are connected.

☛ For more details on using the Web platform for HOPEX solutions, see the **HOPEX Common Features** guide.

Presenting the IT Strategist desktop

All **HOPEX** users have access to the following panes:

- **Home** and **List of Tasks** that are common to all **HOPEX** solution users.
- **Ideation** and **Transformation** used to manage project portfolios and access specific reports if your license allows.

☛ For more information on project portfolio management, see the "Managing projects" section in the **HOPEX Common Features** guide.

- **Reports**: accesses all reports, improving understanding of terms and their use.

In addition to the panes offered in standard mode to all **HOPEX IT Strategy** desktop users, the IT strategist has access to the following panes:

The Vision pane

The **Vision** pane provides access to the following menus.

- **Motivation**, to describe the change drivers and assess them within the framework of strategic assessments;
- **Strategic planning**
 - **Enterprise Strategic View** tree to display the enterprise stages and the Ends and Means of action,
 - **Enterprise Architecture View** tree to display the enterprise stages and the business capabilities maps, the business function architecture environments and the connected solution environments building blocks.
 - Several dedicated reports.
- **Inventories**, to access the main objects.

The Logical Application Architecture pane

The **Logical Application Architecture** pane provides access to the following menus:

- **Logical Application Architecture** to display the logical application architectures tree view as well as several reports;
- **Logical Application Architecture Inventories** provides access to the main objects connected to **logical applications**.

The Application Architecture pane

The **Application Architecture** pane provides access to the following menus:

- **Application Architecture** to display the application architecture hierarchy and their breakdown by stage;
- **Inventories** of main **application** objects.

The Technical Architecture pane

The **Technical Architecture** pane provides access to the following menus:

- **Technical Architecture** to display the tree of the Resource Architecture Environments and their breakdown by stage;
- **Inventories** of main **technical** objects.

Presenting the functional administrator desktop

➤ For more details on the functional administrator desktop of the solution, see the "Presentation of the function administration desktop" section in the **HOPEX Business Architecture** guide.

The HOPEX IT Strategy method

The method embedded in the **HOPEX IT Strategy** solution is identical to the method embedded in the **HOPEX Business Architecture** solution; it is, however, limited to IT transformations.

➤ For more details, see the "**HOPEX Business Architecture** method" section in the **HOPEX Business Architecture** guide.

Starting with HOPEX IT Strategy

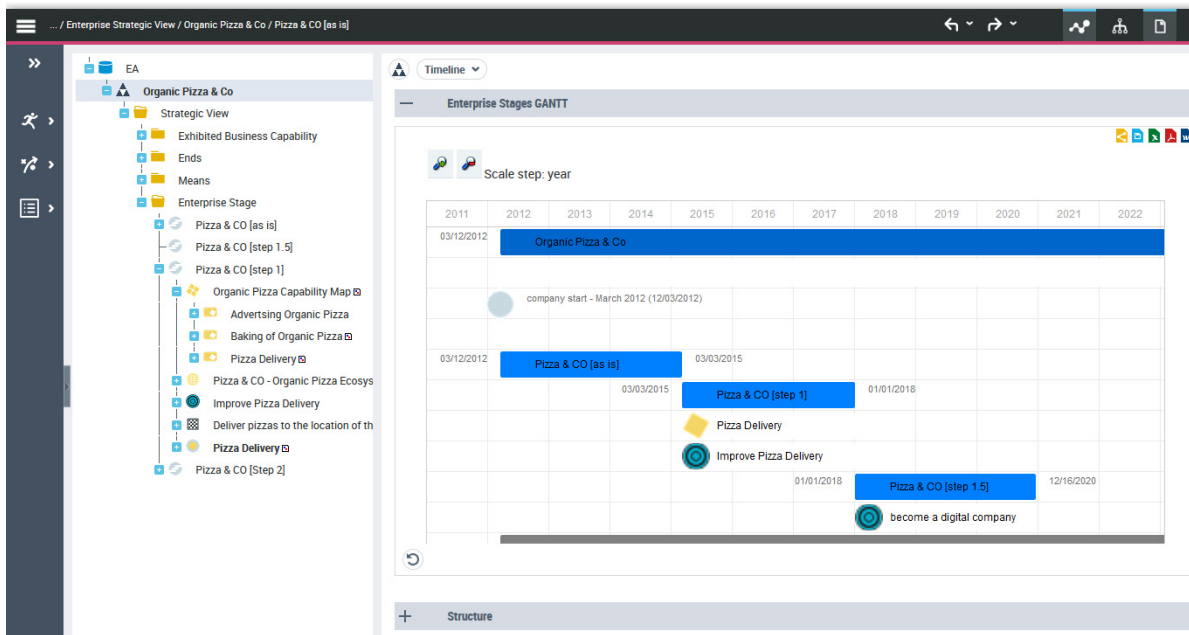
The **HOPEX IT Strategy** functional administration solution is identical to that of the **HOPEX Business Architecture** solution.

Nonetheless, if you have only the **HOPEX IT Strategy** key, you can only build **IT Transformations**.

➤ For more details on the **HOPEX Business Architecture** functional administration, see the " **and HOPEX Business Architecture** functional administration" chapter in the **HOPEX Business Architecture** guide.

MANAGING IT ARCHITECTURE TRANSFORMATION WITH HOPEX IT STRATEGY

The **HOPEX IT Strategy** solution helps you build the roadmap for your transformation project. The transformation roadmap is presented in the form of a Gantt chart.



For more information on Gantt diagrams, see chapter "Using Gantt Charts" in the **HOPEX Business Architecture** guide.


The Gantt chart associated with the enterprise's roadmap is generated to highlight the objectives covered, or to be achieved, as well as business capabilities over time.

Regular adjustment of the transformation project's roadmap requires the following steps:

- Identifying and Assessing Motivations;
- Building and Adjusting the Transformation Roadmap;
- Identifying Strategic Transformation Elements.


Identifying and Assessing Motivations


It is the *stakeholders* who identify the motivations of the transformation project and link them to the elements of the enterprise architecture.

 A stakeholder is an internal or external person or person group with a defined role in the enterprise.

There are various types of drivers:

 A business driver is an expectation expressed by a client, a partner or provider with respect to the enterprise.

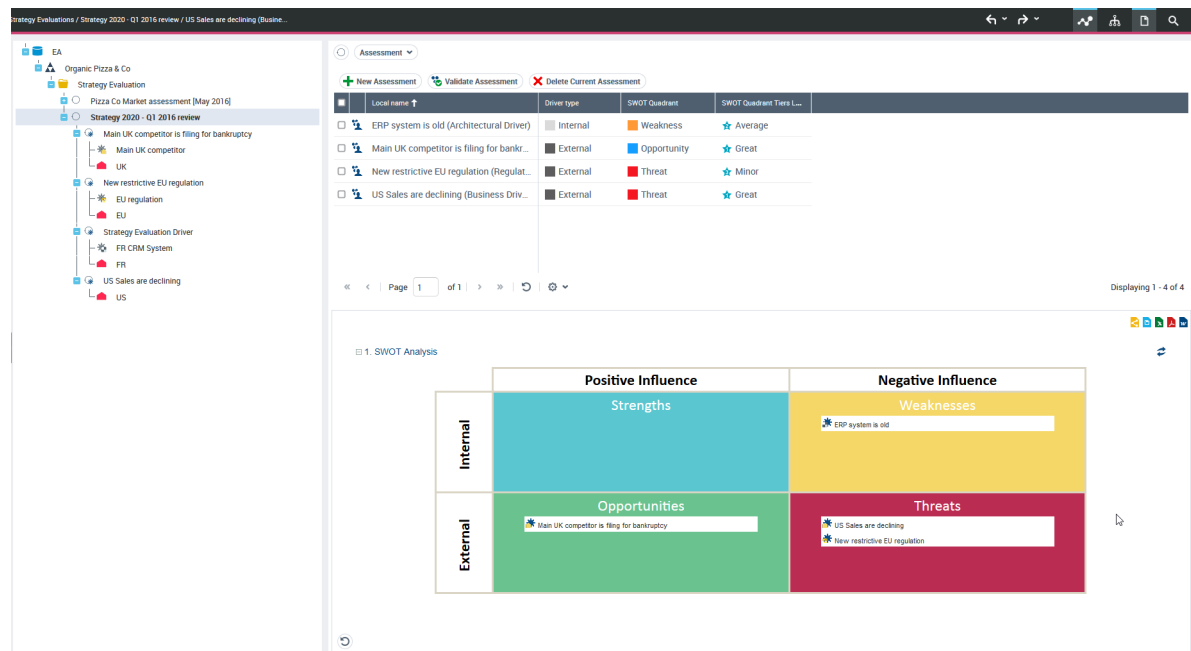
 A regulatory driver is guided by a change in the regulation framework to which it makes reference.

 An architectural driver is guided by a specific characteristic or an internal architectural building block. This characteristic can represent a strength or a weakness

 For more details on transformation drivers, see the "Handling transformation drivers" chapter in the **HOPEX Business Architecture** guide.

The **SWOT** assessment (Strengths, Weaknesses, Opportunities, Threats or Forces) of drivers is possible within the framework of a strategy assessment.

This assessment makes it possible to identify the relevant transformation objectives periodically throughout the duration of the transformation project.



The screenshot shows the HOPEX Business Architecture software interface. On the left, a tree view displays the assessment structure, including 'Organic Pizza & Co', 'Strategy Evaluation', 'Pizza Co Market assessment [May 2016]', and 'Strategy 2020 - Q1 2016 review'. The main area shows a table of drivers and a SWOT matrix.

| Local name | Driver type | SWOT Quadrant | SWOT Quadrant Tiers L... |
|--|-------------|---------------|--------------------------|
| ERP system is old (Architectural Driver) | Internal | Weakness | Average |
| Main UK competitor is filing for bankrupt... | External | Opportunity | Great |
| New restrictive EU regulation (Regulat... | External | Threat | Minor |
| US Sales are declining (Business Driv... | External | Threat | Great |


Below the table, a SWOT Analysis matrix is displayed:


| | Positive Influence | Negative Influence |
|----------|--|--|
| Internal | Strengths | Weaknesses ERP system is old |
| External | Opportunities Main UK competitor is filing for bankruptcy | Threats US Sales are declining New restrictive EU regulation |

 For more details on strategy assessment, see chapter "Using strategy assessments" in the **HOPEX Business Architecture** guide.

Building and Adjusting the Transformation Roadmap

From an enterprise or an enterprise stage, you can define enterprise sub-stages.

 *An enterprise is a purposeful undertaking, an effort conducted by one or more organizations, aiming at delivering goods and services, in accordance with the enterprise mission in its changing environment. In the course of its development, the enterprise must adapt to its environment and establish the transformation objectives and goals to be achieved as well as the strategic action plans used to achieve these objectives. The development and achievement of the different adaptation and transformation stages can lead to a modification of the organization's boundaries. This requires the implementation of an integrated team, under the responsibility of a governing body, to involve the stakeholders in the transformation.*


 *An enterprise stage is a past, current or future stage of an enterprise.*

 *For more details on transformation plans, see chapter "Drawing up the roadmap" in the **HOPEX Business Architecture** guide.*

Enterprise sub-phases can be defined and positioned according to enterprise events, such as mergers and acquisitions, and the introduction of new products or technologies to the market.

Defining enterprise stages

An enterprise is itself an *enterprise stage*; it is therefore possible in **HOPEX IT Strategy** to define business capabilities and enterprise models for courses of action directly at the level of the root enterprise, and refine the iterative roadmap drill down into the subsequent stage levels.


 *An enterprise stage is a past, current or future stage of an enterprise.*

 *For more details on enterprise stages, see the "Defining enterprise stages" chapter in the **HOPEX Business Architecture** guide.*

An *enterprise stage* is defined by a number of components.

- A business capability map, which contains the capabilities valid for the current enterprise stage;
- A logical application system environment, which contains the elements that define the enterprise model (operating model) for the current stage:
 - the definition of the ecosystem of the enterprise (interactions with partners),
 - logical application architectures
 - functionalities.
- The solution building block environments that depend on product licenses used.

Defining the enterprise and its events

 *An event represents a fact or an action occurring in the system, such as updating client information. It is managed by a broker. An application indicates that it can produce the event by declaring that it publishes it. If an application is interested in an event, it declares that it subscribes to the event.*

A basic **enterprise** is made up of the following elements:

- a start enterprise event;
 ➤ The start event can be positioned arbitrarily at the beginning of the current year, for example.
- an end enterprise event;
 ➤ The end event can be positioned with an analysis time frame (e.g.: year $n+5$, year $n+10$)
- a current ('As-Is') enterprise stage that holds the currently deployed business capabilities map, the business architecture environment and the solution building blocks;
 ➤ The end event of this stage is the intermediate event that defines the 'pivot' transformation benchmark beyond which you are in the 'target' stage
- a target ('To-Be') enterprise stage that holds the target business capability map, the business architecture environment and the target solution building blocks.
 ➤ The start date is the end pivot event of the previous ('As-Is') stage.
 ➤ For more details on enterprise stages, see chapter "Managing enterprise events" in the **HOPEX Business Architecture** guide.

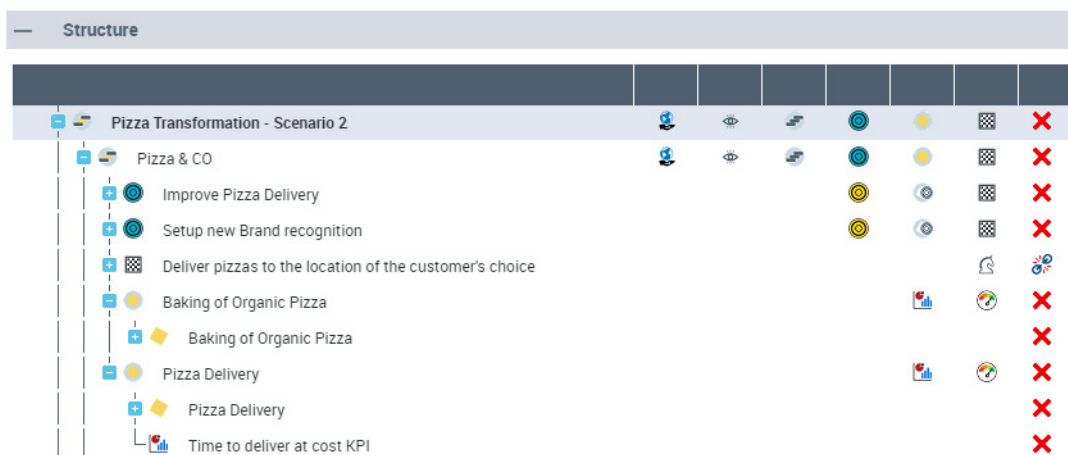
Identifying Strategic Transformation Elements

This step consists of identifying the strategic elements that meet the transformation drivers and that are positioned on the roadmap at each of the enterprise's stages.

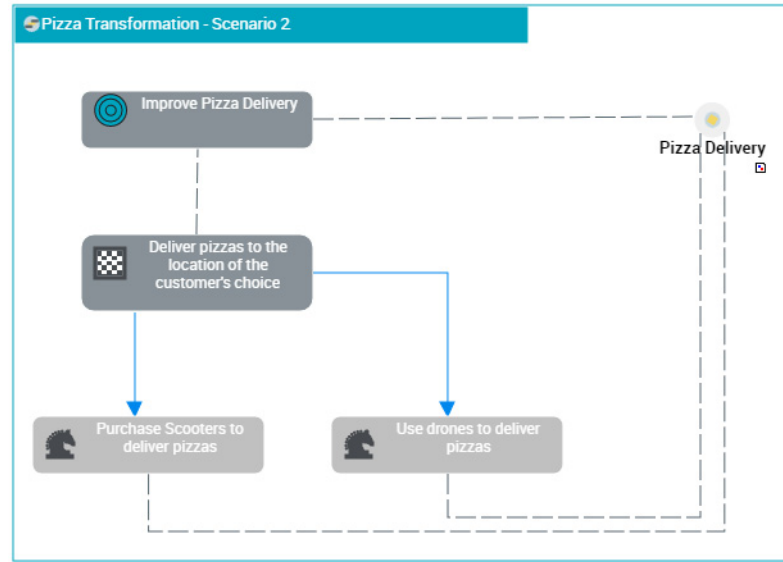
Strategic elements are classified in the following categories:

- Ends, see: [Identifying the transformation ends](#),
- Means, see: [Defining Means](#).
- the exhibited business capabilities, see: [Managing exhibited business capabilities](#).

The strategic elements can be presented in the form of a tree.



An enterprise stage strategy diagram is used to describe the links between the strategic elements (missions, goals, strategies, tactics and exhibited business capabilities).



➤ For more information on this diagram, see the "Building an enterprise stage strategy diagram" chapter in the **HOPEX Business Architecture** guide.

Business capabilities can be placed within stages, to achieve new capabilities or strengthen existing capabilities.

ASSESSING CAPABILITIES WITH HOPEX IT STRATEGY

The **HOPEX IT Strategy** solution helps you define and assess *business capabilities*, of your IT architecture.



.A business capability is a set of features that can be made available by a system (an enterprise or an automated system).

In order to assess your business capabilities you must carry out the following procedures:

- [Describing the Existing Architecture of Business Capabilities](#);
- [Defining the Functionalities Associated with Business Capabilities](#);
- [Assessing a Business Capabilities Map](#).

Describing the Existing Architecture of Business Capabilities

This step is based on the analysis of capabilities carried out with **HOPEX IT Architecture**.



*For more details on managing business capabilities with **HOPEX IT Architecture**, see [Using Business Capabilities with HOPEX IT Architecture](#).*

A *business capability* defines an expected skill.



.A business capability is a set of features that can be made available by a system (an enterprise or an automated system).

For example, to respond to a customer satisfaction objective, the organization must be able to provide services conforming to contractual commitments.

A *business capability map* describes what the enterprise is capable of producing for its internal needs or for meeting the needs of its clients. It is thus based on the main business capabilities of its activity at a given moment.



A business capability map is a set of business capabilities with their dependencies that, together, define a framework for an enterprise stage.

For example, the standard capacity for processing online purchase requests is based on the "Take Customer Call" and "Enter Order" business capabilities.



*For more details on managing a business capability map, see the "Building the business capabilities map" chapter in the **HOPEX Business Architecture** guide.*

Describing a business capability

A business capability is described in more detail by the following elements:

- a more detailed granularity capability breakdown;
- the expected effects of the capability;
- the required functionalities or business skills;
- the dependencies between capabilities (expected effect of one dependent from the result of the other).

☛ For more details on managing a business capability, see the "Describing a business capability" chapter in the **HOPEX Business Architecture** guide.

For example, the business capability that consists of taking "Customer Call" is broken down into a number of business capabilities: "Call analysis", "Identification of the customer".

HOPEX IT Strategy provides a report available detailing the breakdown of capabilities.

☛ For more details on breakdown maps, see the "Business capability breakdown map" chapter in the **HOPEX Business Architecture** guide.

Defining the Functionalities Associated with Business Capabilities

This step is based on the analysis of functionalities carried out with **HOPEX IT Architecture**.

☛ For more details on managing functionalities with **HOPEX IT Architecture**, see [Describing functionalities with HOPEX IT Architecture](#).

To be able to then check that each business capability is correctly implemented by suitable solution building block, you must define the required functionalities.

For example, the "Identify a customer" business capability requires functionalities such as "Search for a customer based on his or her ID".

📖 A functionality is a service required by an org-unit in order to perform its work. This functionality is generally necessary within an activity in order to execute a specific operation. If it is a software functionality, it can be provided by an application.

☛ For more details on business capability skills and functionalities, see the "Defining the skills and functionalities associated with business capabilities" chapter in the **HOPEX Business Architecture** guide.

Assessing a Business Capabilities Map

📖 A business capability map is a set of business capabilities with their dependencies that, together, define a framework for an enterprise stage.

From an enterprise stage or enterprise, it is possible to assess the business capabilities of the business capability map connected to the current stage.

| Local name | Business Value | Capability Efficiency | Capability Effectiveness | Financial Impact |
|--|------------------------|------------------------|--------------------------|------------------|
| Capital Mkt. (Business Capability) | 4 - Limited impact | 2 - Very Efficient | 4 - Slightly Effective | 3 - Moderate |
| Client Facing Common Proc. (Business Capability) | 5 - Negligible impact | 2 - Very Efficient | 3 - Somewhat Effective | 4 - High |
| Common Processing (Business Capability) | 2 - Noticeable impact | 4 - Slightly Efficient | 2 - Very Effective | 4 - High |
| Compliance (Business Capability) | 4 - Limited impact | 2 - Very Efficient | 4 - Slightly Effective | 3 - Moderate |
| Data (Business Capability) | 5 - Negligible impact | 2 - Very Efficient | 3 - Somewhat Effective | 4 - High |
| Finance (Business Capability) | 2 - Noticeable impact | 4 - Slightly Efficient | 2 - Very Effective | 4 - High |
| Internal Audit (Business Capability) | 1 - Significant impact | 2 - Very Efficient | 1 - Extremely Effective | 2 - Low |

For more details on strategy assessment, see chapter "Using strategy assessments" in the **HOPEX Business Architecture** guide.

| Local name | Business Value | Capability Efficiency | Capability Effectiveness | Financial Impact |
|--|------------------------|-------------------------|--------------------------|------------------|
| Capital Mkt. (Business Capability) | 1 - Significant impact | 6 - Future Opportunity | 3 - Somewhat Effective | 3 - Moderate |
| Client Facing Common Proc. (Business Capability) | 2 - Noticeable impact | 4 - Slightly Effective | 5 - Not Effective | 5 - Very High |
| Common Processing (Business Capability) | 2 - Noticeable impact | 1 - Extremely Effective | 5 - Not Effective | 5 - Very High |
| Compliance (Business Capability) | 2 - Noticeable impact | 2 - Very Effective | 5 - Not Effective | 5 - Very High |
| Data (Business Capability) | 2 - Noticeable impact | 3 - Somewhat Effective | 5 - Not Effective | 5 - Very High |

DESCRIBING THE TARGET ARCHITECTURE WITH HOPEX IT STRATEGY

Describing a Logical Application System with HOPEX IT Strategy



A partner application system is an application system external to the environment of the described application service. The partner application system can be a service supplier or a service consumer with respect to application system users.

This step is based on the analysis of logical application systems carried out with **HOPEX IT Architecture**.

For more details on modeling logical application systems with **HOPEX IT Architecture**, see [Describing a Logical Application Architecture with HOPEX IT Architecture](#).

Accessing the list of logical application systems with HOPEX IT Strategy

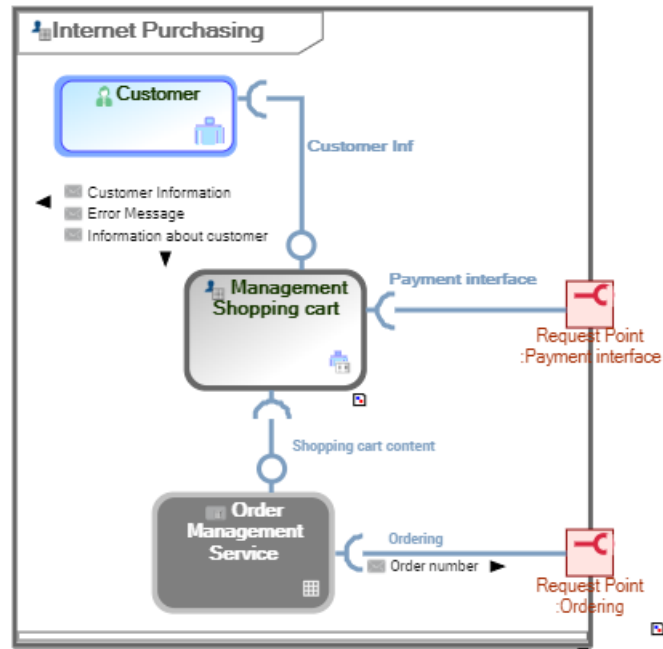
To access the list of logical application systems from the **Logical Application Architecture** navigation pane:

- 1 Select **Logical Application Architecture Inventories > Logical Application System** in the navigation menu.
The list of logical application systems appears.

Describing a Logical Application System with HOPEX IT Strategy

An logical application system diagram describes the interactions between the main components of the system described.

The following diagram describes the structure of the Internet purchasing logical application system proposed to customers.

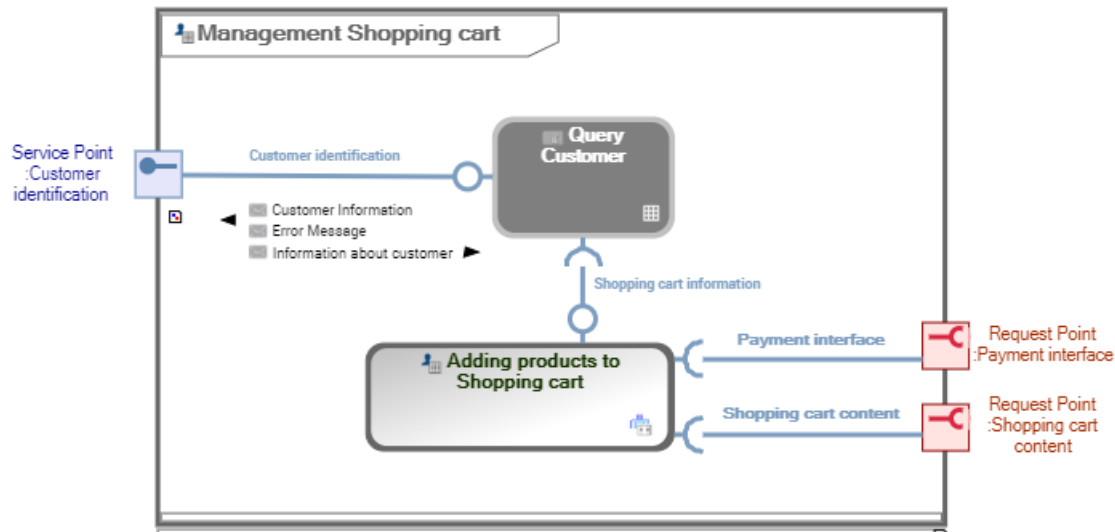


Structure diagram of the "Internet Purchasing" logical application system

Requests made by customers are processed by a "Management Shopping Cart" logical application system. The "Order

Creation" logical application is then used in the context of this architecture.

The structure diagram of the logical application system, responsible for customer shopping cart management, presents two request points for "Booking" or "Ordering".



Structure diagram of the "Management Shopping Cart" logical application system

For more details on a logical application system representation, see [Describing a Logical Application System with HOPEX IT Architecture](#).

Describing Logical Applications with HOPEX IT Strategy

An application system is an assembly of other application systems, applications and end users interacting with application components to implement one or several functions.


For more details on logical applications, see [Describing Logical Applications With HOPEX IT Architecture](#).

Accessing the list of logical applications with HOPEX IT Strategy

To access the list of logical applications from the **Logical Application Architecture** navigation pane:

- 1 Select **Logical Application Architecture Inventories > Logical Applications** in the navigation menu.
The list of logical applications appears in the edit area.

Describing the Logical Application System Environment with HOPEX IT Strategy

 A logical application system environment presents a logical application system use context. It describes the interactions between the logical application system and its external partners, which allows it to fulfill its mission and ensure the expected functionalities.

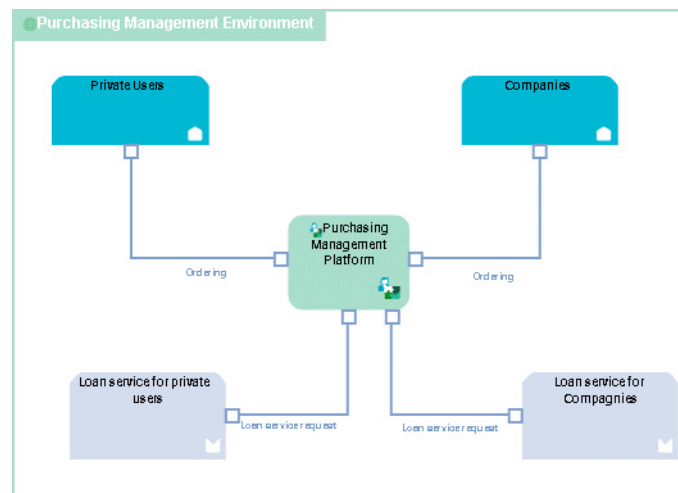
To access the list of logical application systems from the **Logical Application Architecture** navigation pane:

- 1 Select **Logical Application Architecture Inventories > Logical application system environment** in the navigation menu.
The list of logical application system environments appears in the edit area.

A logical application system diagram describes the interactions between the main internal components of the environment described and the external components.

In this example, purchase requests are formulated by private users or by companies in different contractual conditions.

The "Purchasing Requests Processing" application system offers a loan service to its clients within the context of payment management.




Environment diagram for the "Purchasing Requests Processing" application system.

 For more details on a logical application system environment representation, see [Using the Environment Structure Diagram of a Logical Application System](#).

Describing Implementation of a Business Capabilities Map

This involves connecting the *business capability*, which corresponds to what we know how to do or what we want to do and which represents the goal (*the end*) to be achieved, to a way of achieving that which is represented by a *logical application* or a *logical application system* at a conceptual level, that is, upstream of organizational and technical choices.

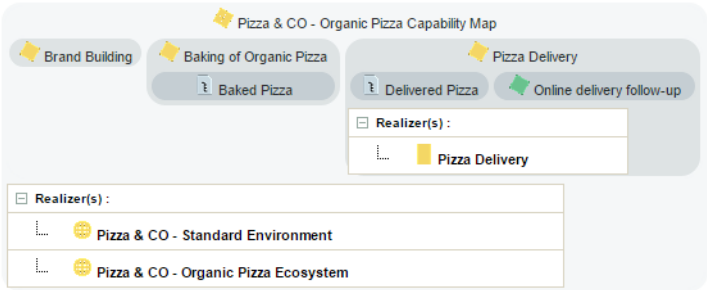
 A logical application system is an assembly of other application architectures, logical applications and end users, interacting with application components to implement one or several functions.

By constructing the *business capability map* on the one hand and the *logical application system environment* on the other hand, you can check that the business capabilities are implemented by the logical applications.


 For more details on the logical applications associated with business capabilities, see "Describing component implementation" chapter in **HOPEX Business Architecture** guide.

HOPEX IT Strategy provides a report that presents the result of the implementation of business capabilities by *Logical Application* or *Logical Application System*.

1. Capability Map Report



Example of business architecture breakdown report

 For more details on the breakdown of business capabilities, see the "Breakdown map of business capabilities" chapter in the **HOPEX Business Architecture** guide.



USING IT ARCHITECTURE DIAGRAMS



This chapter explains how to build the main types of IT Architecture diagrams.

The table below draws up the list of solutions for which several descriptions are proposed for different types of objects.


| Object type | Structure diagram | Flow Scenario | technical architecture |
|-------------------------|---|---|---|
| Application | HOPEX Application Design HOPEX IT Architecture | HOPEX Application Design HOPEX IT Architecture | HOPEX Application Design HOPEX IT Architecture |
| Application System | HOPEX IT Architecture | HOPEX IT Architecture | HOPEX IT Architecture |
| Application Environment | HOPEX Application Design HOPEX IT Architecture | HOPEX Application Design HOPEX IT Architecture | HOPEX Application Design HOPEX IT Architecture |
| IT service | HOPEX Application Design HOPEX IT Architecture | HOPEX Application Design HOPEX IT Architecture | HOPEX Application Design HOPEX IT Architecture |
| Micro-service | HOPEX Application Design HOPEX IT Architecture | HOPEX Application Design HOPEX IT Architecture | HOPEX Application Design HOPEX IT Architecture |

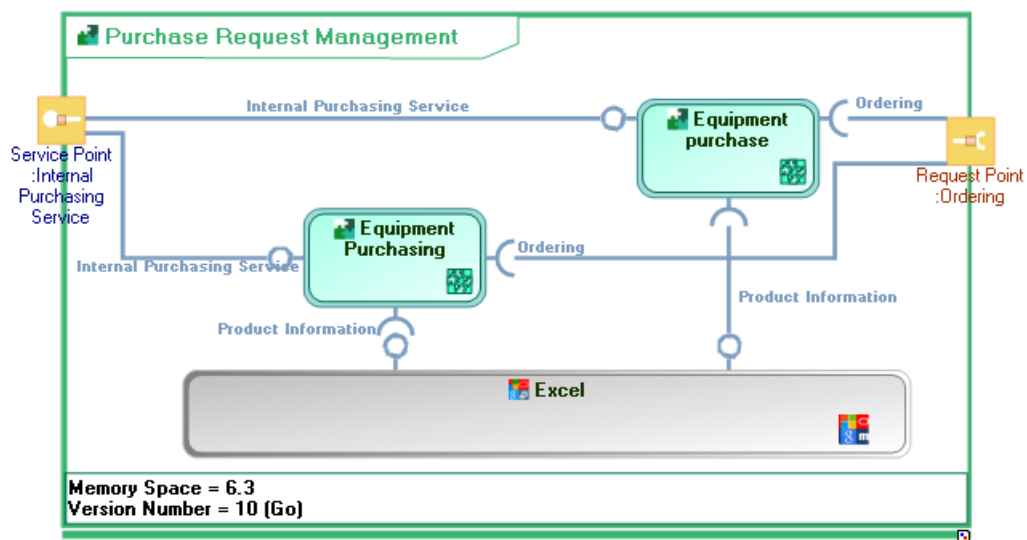
- ✓ ["Creating a structure diagram", page 156;](#)
- ✓ ["Describing a scenario of flows", page 158;](#)
- ✓ ["Describing a Technical Architecture", page 164.](#)

CREATING A STRUCTURE DIAGRAM

With **HOPEX IT Architecture**, the components of an object and their exchanges are described in a structure diagram.

Creating an Application Structure Diagram

 An application structure diagram graphically shows first level components of an application, the access points (service point and request point) and the connections between components.




The purchase request management application, which is used only for internal purchases, is based on two specialized service applications: one for office supplies and the other for equipment. Both application services use Microsoft Excel.

The components of an Application Structure Diagram

An Application Structure Diagram includes:

- **IT services** which represent the IT services used and deployed with the application.


In the example, this is the office supply application.

 An IT service is a software component of an application, that can't be deployed alone and that realizes a sub-set of the functionalities of

this application either for end users of this application or inside the application (or another application). This includes batch programs.

- **micro-services** which represent the services used independently of the application.


In the example, this is the Microsoft Excel application.

 A micro-service is a software component that can be deployed autonomously, but which does not directly provide an end user service. It can interact with other application services, applications or application systems. This is a deployable software component that uses software technologies. For example: an authentication service, a PDF file printing service.


- request and service points

 For more details, see "[Describing Service and Request Points](#)", page 169.

- **interactions** between components.

 An interaction represents a contract established in a specific context between autonomous entities that are internal or external to an enterprise. These entities can be enterprise org-units, applications, activities or processes, as well as external org-units. The content of this contract is described by an exchange contract.

- **physical data stores** used by the application.

 For more details, see "[Managing Data](#)", page 96.

Adding an application service to an application structure diagram

To describe that an application uses an application service, go to:

1. In the object toolbar of the application structure diagram, select **Application Service** and click in the frame of the application described. An addition dialog box asks you to select the **application service** used.
2. Select an existing application service
3. Click **OK**.
The application service appears in the diagram.

DESCRIBING A SCENARIO OF FLOWS

The scenario of flows diagram describes the flows exchanged between the system elements represented.

With **HOPEX**, two types of diagrams are proposed.

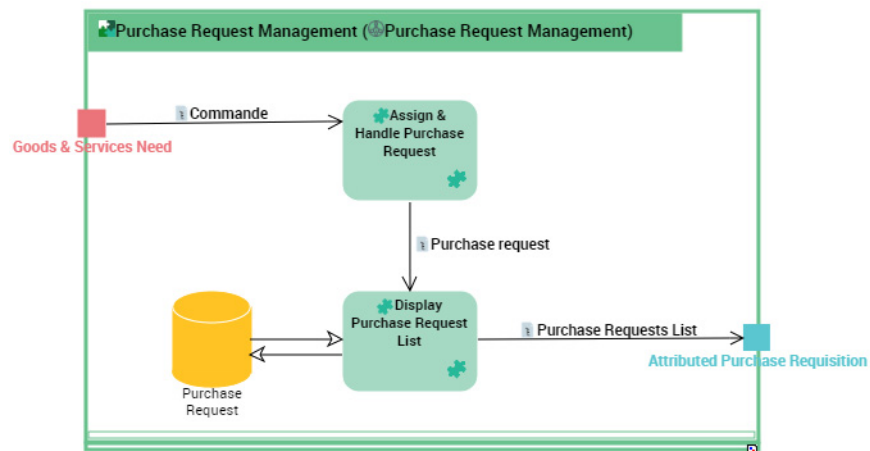
- The *Scenario of flows diagrams* that describe the flows exchanged in different use scenarios of the object described.
- The *Scenario Sequence Diagrams* that describe the chronology of the flows exchanged in different use scenarios of the object described.

Using a Scenario of Application Flows Diagram

An Application Flow Scenario Diagram can be built for an application environment, an application, an Application System, an IT service or a micro-service.

The flow scenario diagram below describes the "Spare Parts Purchasing" application.

The Scenario of Application Flows below describes the exchanges between the "Purchase request management" components.



Example of a Scenario of Application Flows for "Managing Purchase Orders".

In a scenario of application flows diagram, the elements represented are:

- IT services,
- micro services,
- stores of internal or external application data,
- input or output application ports.

The interactions offered between these elements:


- application flows that carry a content,
- application flow channels that group a number of application flows on a single link,
- application data channels that represent the interactions between the application data stores.

Creating a Scenario of Application Flows diagram

To create a scenario of application flows:

1. Right-click the application and select **New > Scenario of Application Flows**.
2. In the window for choosing the diagram type, select **Scenario of Application Flows**.


Adding an IT service to the scenario of application flows

 *An IT service is a software component of an application, that can't be deployed alone and that realizes a sub-set of the functionalities of this application either for end users of this application or inside the application (or another application). This includes batch programs.*

To add an **IT service**:


1. In the objects toolbar of the scenario of application flows, click **Application**.
2. Click in the described application frame.
An addition window box prompts you to choose the **application service** implemented (for example "Customer management").
3. Select the application service required and click **OK**.
The application service appears in the diagram.

You can add a micro-service in the same way.


 *A micro-service is a software component that can be deployed autonomously, but which does not directly provide an end user service. It can interact with other application services, applications or application systems. This is a deployable software component that uses software technologies. For example: an authentication service, a PDF file printing service.*

Managing application flows in a scenario of application flows

Creating an application flow with content

 *An application flow represents the circulation of information between applications or within an application. An application flow can carry a content.*

The application flows exchanged between the IT services, the micro-services or the Application ports of a scenario of application flows are associated with a **content**.

 *The content designates the content of a message or an event, independent of its structure. This structure is represented by an XML schema linked to the content. A content may be used by several messages, since it is not associated with a sender and a destination. There can be only one content per message or event, but the same content can be used by several messages or events.*

You must directly specify the *content* of an *application flow* directly on flow creation.

To create the *application flow*:

1. Click the reel in the objects toolbar of a scenario of application flows.
2. Click the first object representing the sender of the flow and, holding the mouse button pressed, draw a link to the object receiving the flow.
The **Application Flow Creation** dialog box opens.
3. In the **Content** drop-down list, select the content you wish to associate with the flow.
The application flow is displayed with its content in the diagram.

Creating an application flow channel



An application flow channel is used to graphically group a number of application flows into a single flow.

To create an application flow channel, you must first create the channel and then link the application flows that it groups.

To create an *application flow channel*:

1. In the objects toolbar of the scenario of application flows, click **Application Flow Channel**.
2. Click the first object in communication and, holding the mouse button pressed, draw a link to the other object.
The application flow channel appears in the diagram.

To connect the application flows to the *application flow channel*:

1. Open the **Characteristics** properties page of the application flow channel.
2. In the **Grouped Flow** section, click **Connect**.
A selection dialog box opens and presents the list of the ungrouped application flows of the scenario of application flows.
3. Select the flows that you want to group and click **OK**.
The application flow content appears with an arrow that marks the direction of the flow.

Adding an application data store to the scenario of application system flows



An application data store materializes the usage of data in the context of a software component (for instance an application). An application data store provides a mechanism to retrieve or update information stored outside of the current software component.

➡ For more information on managing data stores, see ["Managing Data", page 96](#).

A data store can be local or external to the application.

To add, for example, a local application data store to an scenario of application flows

1. In the scenario objects toolbar, click **Local Application Data Store**.

2. Click in the described application frame.
An addition window prompts you to choose the **data area** that represents the physical structure that will concretely support the application data store.



A data area represents a restricted data structure dedicated to the description of a software Data Store. It is made of classes and/or data views and can be described in a Data Area Diagram.

➤ *For more information on data areas, see chapter "Logical and application data areas" in the **HOPEX Information Architecture** guide.*

3. Select the existing **Data Area** that interests you.
4. Click **OK**.
The local application data store appears in the diagram with the name of the physical data domain selected.

Creating an application data channel

The applications, the application systems and the micro-services can have read or right access to a local or external application data store.

To create an application data channel that represents a reading access:

1. In the diagram objects toolbar, click **Application Data Channel**.
2. Draw a link between the application data store and the object that reads the data.

An application data channel automatically appears in the scenario.

➤ *To create a link with write access, you must draw a link between the object that reads and the application data store.*

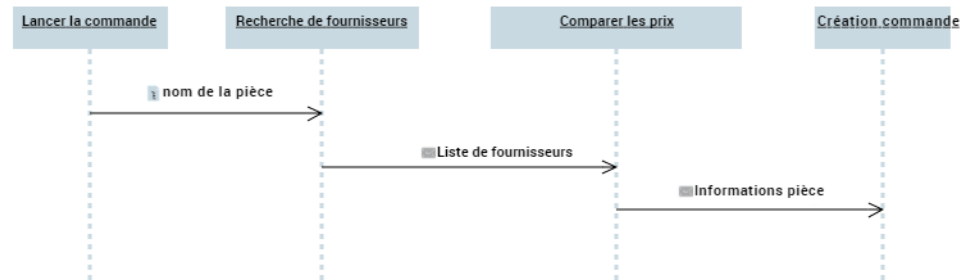
Using a Scenario Sequence Diagram

For each use context, you can create scenario sequence diagrams. A scenario sequence diagram presents the same exchanges between system elements, highlighting their chronology. The elements in the sequence scenario are represented in the diagram by lines.

A scenario sequence of flow diagram contains:

- Lines which define interaction participants: instances of applications, services or interfaces.
- Different types of messages exchanged between participants.
- Advanced functions that enable concise description of several execution sequences.

☛ For more information on sequence diagrams, see the "**HOPEX UML**" guide.



This diagram describes the operation of the "Order Unreferenced Parts" use case :

When a purchase request is entered in the user interface, the name of the part is received by the "Find Suppliers" service, which draws up the list of suppliers offering the requested part.

The "Compare Prices" service looks for the lowest-priced product and sends information to the "Order Amount Calculation" service.

When the order amount has been established, a final "Issue Purchase Order" service sends the order via the interface.

Creating an application environment scenario sequence diagram

To create an application environment scenario sequence:

1. Right-click the application environment and select **New > Application Environment Scenario Sequence Diagram**.
2. In the window for choosing the diagram type, select **Application Environment Scenario Sequence Diagram**.

Instances of applications, IT services or interfaces

Depending on whether the diagram describes a user interface, an application or a, IT service, the interaction scenario diagram describes messages exchanged between application instances, *IT service* instances and *user interface* instances.

📖 A Human-Machine Interface enables definition of a screen of an application or an IT service.

📖 An IT service is a software component of an application, that can't be deployed alone and that realizes a sub-set of the functionalities of this application either for end users of this application or inside the

application (or another application). This includes batch programs.

To create an application service instance for example:

1. Click the **Application Service** button in the toolbar.
2. Click on the diagram.
The **Add Application Service** dialog box appears.
3. Click the arrow to the right of the **Name** field and select **Connect Application Service** in the drop-down list.
The list of application services accessible from the current library appears.
4. Select the IT service you require.
5. Click **OK**.
The application service instance appears in the diagram.

Message instance

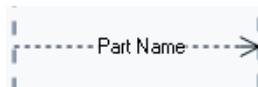
Message instances define the data exchanged between application instances, application services and the interfaces. The sequence described in the flow scenario sequence diagram indicates the message sending order.

Message instances displayed in an interaction scenario diagram correspond to messages owned by the application that have been previously defined in another diagram.

To create a message instance:

1. Click the reel in the toolbar.
2. Click the dotted line under the first object and, holding the mouse button down, draw a line to the second object.
3. Release the mouse button.

The message instance exchanged between the two objects is drawn.

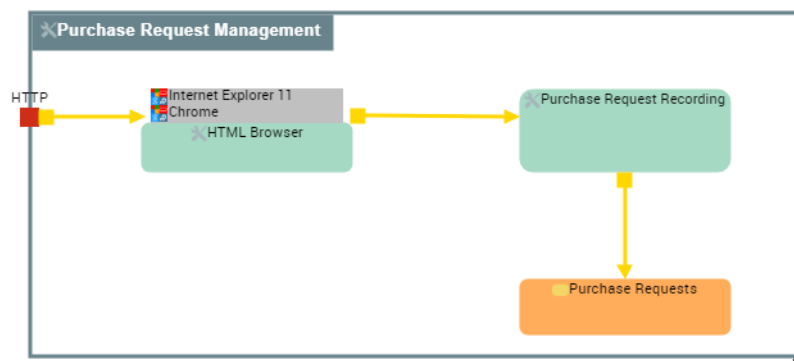


DESCRIBING A TECHNICAL ARCHITECTURE

An *Technical Architecture* is used to represent the Technical Architectures and the Technical Data Areas of the described object as well as the techniques used for their communications.

The application technical architecture below presents the technical areas used by the purchase request management application.

The technical communication line is based on http.



Technical Architecture Example



An application technical architecture describes one of the configurations possible for application deployment. It describes how the different technical areas of the application are connected to each other and the technologies and the communication protocols that they use. An application can have a number of possible technical architectures (E.g.: autonomous installation, horizontal or vertical deployment, etc.)

Creating an Application Technical Architecture Diagram

Elements represented in an Application Technical Architecture diagram are:

- *application technical areas,*



An application technical area represents an organizational element of an application according to technical criteria. For example, this can be the user interface or a process. Each application technical area is associated with one or more technologies. The deployment of several application technical areas is necessary for the application to be operation.

- *Technical Data Areas,*



A data technical area represents an organizational element of an application used to access the data necessary for the operation of this application. Each application technical area is associated with one or more technologies (E.g.: Oracle 12, SQL Server 2012, etc.). A data

technical area can allow access to one or more data stores.

- **technical input** and output ports,



A technical server port is a point used to open communications with a technical architecture or an application technical area in compliance with a particular communication protocol (SMTP, HTTP, etc.).

- **Technical Communication Lines.**



A technical communication line represents a technical connection between architectures or application technical areas through client and server ports. Client technical port of an architecture or a technical area requires opening the communication line to server technical port of the other area or technical architecture.

Adding an application technical area to an application technical architecture diagram



An application technical area represents an organizational element of an application according to technical criteria. For example, this can be the user interface or a process. Each application technical area is associated with one or more technologies. The deployment of several application technical areas is necessary for the application to be operation.

To create an **application technical area**:

1. In the objects toolbar of the application technical architecture, click **Application Technical Area**.
2. Click in the described application frame.
An addition window prompts you to choose the **Application Technical Area** that you wish to use.
3. Select the application technical area and click **OK**.
The application technical area appears in the diagram.

You can add data technical areas in the same way.



A data technical area represents an organizational element of an application used to access the data necessary for the operation of this application. Each application technical area is associated with one or more technologies (E.g.: Oracle 12, SQL Server 2012, etc.). A data technical area can allow access to one or more data stores.

Defining the software technologies used by an application technical area



A software technology is a basic component necessary for operation of business applications. Software technologies include all basic software such as: application server, electronic mail server, software components for presentation, data entry, storage, business information sharing, operating systems, middleware, navigators, etc.

To specify the **software technologies required** for an **application technical area**:

1. Open the **Characteristics** property page of the **Application Technical Area** that interests you.
2. In the **Software Technologies Required** section, click **Connect**.
In the selection dialog box, select the **Software Technologies** that you want to use.
The software technologies selected appear in the icon for the application technical area.

Creating a technical communication line



A technical communication line represents a technical connection between architectures or application technical areas through client and server ports. Client technical port of an architecture or a technical area requires opening the communication line to server technical port of the other area or technical architecture.

The communication techniques between the application technical areas and the data areas can be described by technical communication lines.

To create a technical communication line, you must first create the channel and then specify the communication protocols that are used.

To create a technical communication link:

1. In the diagram objects toolbar, click Technical Communication Line.
2. Draw a line between the two communicating objects.
The technical communication line appears in the architecture.

To connect the protocols to the *technical communication line*:

1. Open the **Characteristics** properties page of the technical communication line.
2. Click the **Connect** button.
A selection dialog box opens displaying the list of communication formats.
3. Select the formats that you want to use and click **Connect**.

DESCRIBING DATA EXCHANGES




This chapter explains how to describe exchange contracts between the components of a business or IT architecture.


- ✓ ["Managing Interactions", page 168;](#)
- ✓ ["Describing Exchanges", page 173;](#)
- ✓ ["Describing Exchange Contracts", page 176.](#)


MANAGING INTERACTIONS

An *Interaction* represents the exchange of information between architecture components.

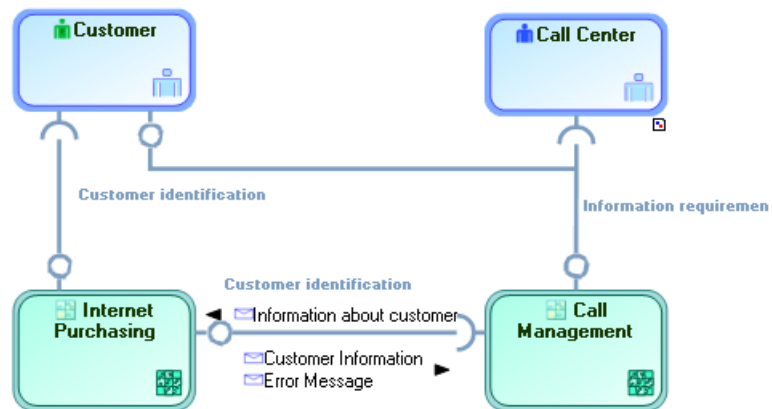
 An interaction represents a contract established in a specific context between autonomous entities that are internal or external to an enterprise. These entities can be enterprise org-units, applications, activities or processes, as well as external org-units. The content of this contract is described by an exchange contract.

Content of an interaction is described by an *exchange contract*.

 An exchange contract is a model of a contract between organizational entities. This contract is described by exchanges between an initiator role and one or several contributor roles.

 For more details on exchange contract concepts, see ["Describing Exchange Contracts"](#), page 176.

In a "Purchasing Requests Processing" application system structure diagram, two exchange contracts are used by different interactions.





Interactions in the "Purchasing Requests Processing" application system structure diagram

The clients must be identified before entering an order. They can enter orders directly from an eCommerce application or by using a Call Center. The Call Center uses the "Call Management" application which uses the client identification service offered by the "eCommerce Purchasing" application.


Creating an Interaction


To create an interaction:

1. In the objects toolbar for a diagram, click **Interaction** .
2. Click the entity requesting the service and draw a link to the entity providing the service.
3. In the add interaction dialog box, specify the exchange contract you wish to use.
 You can also create a new exchange contract. For more details, see ["Creating an Exchange Contract from an Interaction", page 178](#).
4. Click **Add**.

Describing Service and Request Points

In a service-oriented architecture, communication is based on access points: *service points* and *request points*.


 A request point is a point of exchange by which an agent requests a service from potential suppliers.


 A service point is a point of exchange by which an agent offers a service to potential customers.

Service points

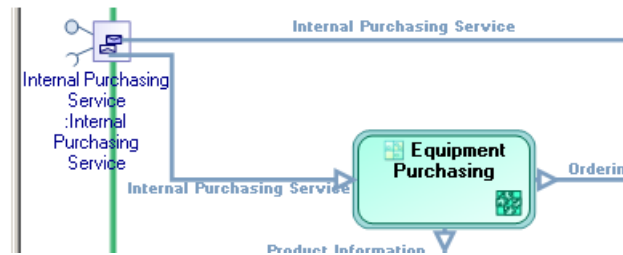
An application system, for example, is created to ensure one or more services. These services are represented by *service points*.

The service is requested according to precise terms defined by an *exchange contract* assigned to the service point.

 An exchange contract is a model of a contract between organizational entities. This contract is described by exchanges between an initiator role and one or several contributor roles.

 For more details on exchange contracts, see ["Describing Exchange Contracts", page 176](#).


Components activated to assure a service are linked to the service point by interactions. If it is necessary to activate several components, you have to create several interactions between the service point and the system components.




In the example presented here, the internal purchasing service is linked to two interactions based on the same exchange contract, representing the activation of the equipment purchasing application or the office supplies purchasing application.


☛ To create a service point, see ["Creating a Service Point or a Request Point", page 171](#).

Request points

A **request point**  enables representation of use of a service external to the described entity.

 A request point is a point of exchange by which an agent requests a service from potential suppliers.

A service point is a point of exchange by which an agent offers a service to potential customers. The service is requested according to precise terms defined by an **exchange contract** assigned to the request point.

 An exchange contract is a model of a contract between organizational entities. This contract is described by exchanges between an initiator role and one or several contributor roles.

☛ For more details on exchange contracts, see ["Describing Exchange Contracts", page 176](#).

Components that issue a request are linked to the request point by an interaction.





In the example, request points represent requests for service executed by the "Adding Products to Shopping Cart" logical application to issue an order or book products.

☛ To create a request point, see ["Creating a Service Point or a Request Point", page 171](#).



Creating a Service Point or a Request Point

The process for creating a *service point* or *request point* is identical.


 A request point is a point of exchange by which an agent requests a service from potential suppliers.

 A service point is a point of exchange by which an agent offers a service to potential customers.

To create a service point:

1. In the diagram insert toolbar, click **Service Point** .
2. Position the object at the edge of the frame of the described object.
A creation dialog box opens.
3. Click the arrow to the right of the **Exchange Contract** field to define the exchange contract enabling activation of this service point, and select, for example, **Connect Exchange Contract**.
A query window opens.
4. Select the exchange contract associated with this service point and click **Connect**.
5. Click **Next**.
A dialog box opens proposing a list of exchange contract roles that can be associated with the service point.
 This dialog box is not proposed if there is only one candidate role that can be associated with the service point.
6. Select the role that interests you and click **OK**.
The service point appears in the diagram.

To change the service point name:

1. Click the name of the service point and press key F2.
2. Enter the new name used when specifying interaction points.
 For more details on interaction points, see ["Describing Service and Request Points", page 169](#).

Defining the Element Interaction Point



The interaction point of an element connects an interaction to one of the components in communication. This specifies:

- on the one hand, the service point, or the request point, that intervenes in the communication
- on the other hand, the role, consumer or supplier represented by the interaction point in the exchange contract.

Characterizing the element interaction point


To modify the properties of the element interaction point:


1. Right-click the interaction beside the communication element.
2. Open the **Characteristics** properties page.

3. Select the **Played Service Role**, that is the role of the exchange contract played by the element interaction point.
 For more details on the roles of an exchange contract, see ["Creating an exchange diagram \(BPMN\)", page 174](#).
4. Select the **Interaction Endpoint Target**, that is the service (or request) point concerned by the interaction.
 For more information on service points or request points, see ["Describing Service and Request Points", page 169](#).
5. Click **OK**.


DESCRIBING EXCHANGES

Content of an interaction is described by an *exchange contract*.

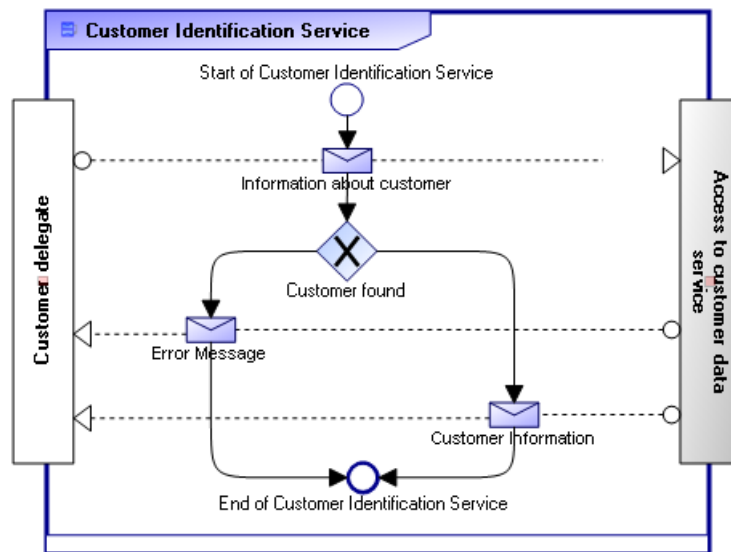
 An exchange contract is a model of a contract between organizational entities. This contract is described by exchanges between an initiator role and one or several contributor roles.

 For more details on exchange contracts, see ["Describing Exchange Contracts"](#), page 176.

An exchange contract is described by a sequence flow of exchanges or exchange contracts.

 An exchange specifies message flow exchanges between two participants.

An exchange diagram describes the sequence flows of an *exchange*.




"Customer Identification Service" Exchange Diagram

The customer identification service protocol begins by sending information enabling identification of the customer. An error message appears if the customer is not found, otherwise customer information is sent (customer identification, status of orders, etc.).

Creating an Exchange

You can create an *exchange* from an exchange contract diagram (BPMN).

To create an *exchange* from an exchange contract diagram (BPMN).

1. Click the **Exchange Use** button  and click in the diagram within the described exchange contract frame.



An exchange use represents the usage of an exchange in another exchange contract.

The Creation of Exchange Use dialog box opens.

2. Click the arrow to the right of the **Exchange Specification** field and select **Create Exchange** in the drop-down list.
The Creation of Exchange dialog box appears.
3. Enter the **Name** of your exchange and click **OK**.
4. In the **From** field, select the exchange contract role described connected to the "Consumer" role of the exchange used.
5. In the **To** field, select the exchange contract role described connected to the Supplier role of the exchange used.
6. Click **Finish**.
7. Click **OK**.
The exchange is automatically created.

Describing Exchanges

Creating an exchange diagram (BPMN)

An *exchange* is described by an exchange diagram presenting the sequence flow of messages exchanged.

To create an exchange diagram:

1. Right-click an **Exchange** and select **New > Exchange Diagram (BPMN)**.

The diagram opens. The exchange frame is positioned and the two roles (Consumer and Supplier) are created.

Creating a message flow with content

You must specify the *message flows* and their *content* exchanged between the two exchange roles.



A message flow represents circulation of information within an exchange contract. A message flow transports its content.



The content designates the content of a message or an event, independent of its structure. This structure is represented by an XML schema linked to the content. A content may be used by several messages, since it is not associated with a sender and a destination. There can be only one content per message or event, but the same content can be used by several messages or events.


To create a message flow and its content:

1. In the exchange diagram, click the **Flow With Content** button.
2. Click the role that represents the message flow sender and, holding the mouse button down, draw a link to the message flow recipient.
The **Creation of Message Flows With Content** dialog box opens.


3. In the **Content** drop-down list, select the content you wish to associate with the flow.
The message flow is displayed with its content in the diagram.

Managing events, gateways and sequence flows


"Start" and "End" **events** are required in the description of the service assured by the exchange contract.


 *An event represents a fact or an action occurring in the system, such as updating client information. It is managed by a broker. An application indicates that it can produce the event by declaring that it publishes it. If an application is interested in an event, it declares that it subscribes to the event.*

In compliance with the BPMN standard, in the object toolbar, several **gateway** types are available to you.

 *Gateways are modeling elements that are used to control how sequence flows interact as they converge and diverge within a process.*

A **sequence flow** is a directional link that represents the chronological organization of the different processing steps.

 *A sequence flow is used to show the order in which steps of an exchange contract will be performed. A sequence flow has only one source and only one target.*

 *For more details on events, gateways and sequence flows, see ["Managing events, gateways and sequence flows", page 175](#)*

DESCRIBING EXCHANGE CONTRACTS

An *Interaction* represents the exchange of information between architecture components.



An interaction represents a contract established in a specific context between autonomous entities that are internal or external to an enterprise. These entities can be enterprise org-units, applications, activities or processes, as well as external org-units. The content of this contract is described by an exchange contract.

Content of an interaction is described by an *exchange contract*.



An exchange contract is a model of a contract between organizational entities. This contract is described by exchanges between an initiator role and one or several contributor roles.

An exchange contract is described by a sequence flow of operations which are represented:

- by *exchange contract use*



An exchange contract use is associated with an exchange contract. It enables representation of complex exchanges.

- or by *exchange use*



An exchange use represents the usage of an exchange in another exchange contract.

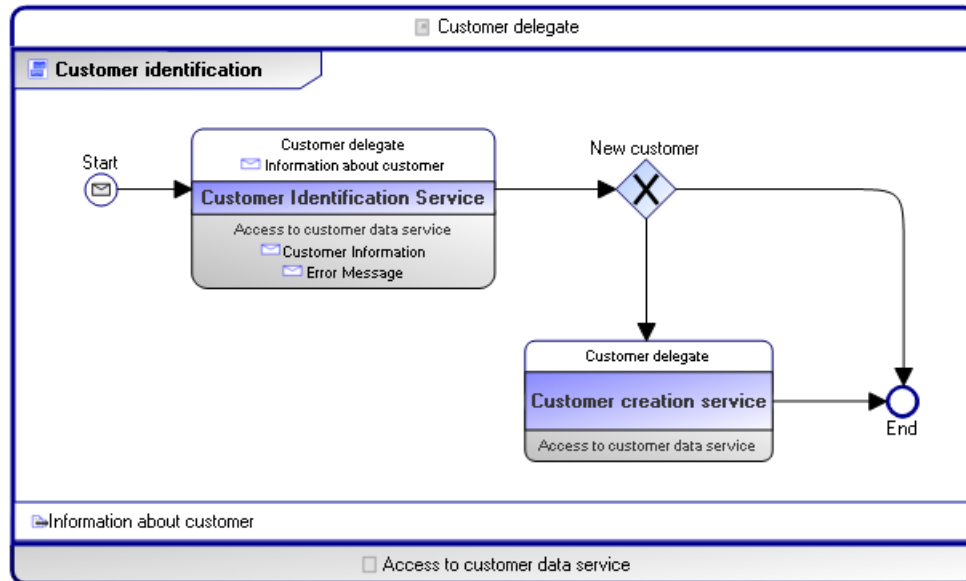


For more details on exchanges, see "[Describing Exchanges](#)", page 173.

Examples of Exchange Contract Diagrams (BPMN)

Exchange contract diagram (BPMN)

The exchange contract diagram associated with the customer identification exchange contract describes, in BPMN formalism, the operations executed.



Exchange Contract Diagram (BPMN) "Customer Identification"

Customer identification protocol starts with a customer identification step. If the customer is found the exchange contract returns customer information, if not, a customer creation exchange contract is activated.

Progress steps are represented by *exchange use*.

An exchange use represents the usage of an exchange in another exchange contract.

Advanced communication exchange contract example

An exchange contract is described by a sequence flow of steps which are represented:

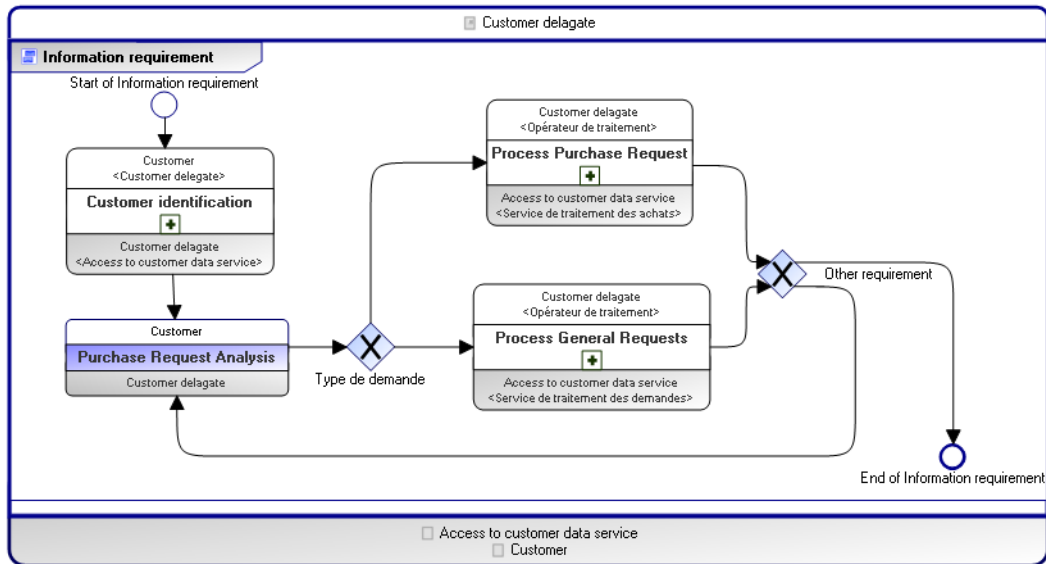
- either by *exchange use*
- or by *exchange contract use*

An exchange contract use is associated with an exchange contract. It enables representation of complex exchanges.

The exchange contract roles, presented at the border of the frame, represent participants:

- customer/supplier, or
- sender/recipient

An exchange can be described by involving more than two participants. In this case, one role is the initiator of the exchange contract and the others are contributors.



"Information Requirement" Exchange Contract Diagram (BPMN)

The "Information Request" exchange contract is used by the call center to take account of a customer request online. There are therefore three participants in this exchange contract: the customer, the IT applications and the customer representative who is the effective requester of the service (in this case the call center).


This exchange contract consists of identifying the customer, then analyzing the request. The request is then processed as a purchase request or as another request if it is an information request for example.

Creating an Exchange Contract from an Interaction

You can also create a new exchange contract:

- from a library,
- from an interaction in a diagram.

To create an exchange contract, in a diagram, from an interaction:


1. In the diagram insert toolbar, click the **Interaction** button. 
2. Draw a link between the two communication entities.
3. In the add interaction dialog box, click the arrow at the right of the **Exchange Contract** box and select **New**.
The **Creation of Exchange Contract** dialog box opens.
4. Enter the name of the exchange contract in the **Name** box.
5. Click **OK**.
The interaction and exchange contract are created.

Creating an Exchange Contract Diagram (BPMN)


An exchange contract is represented by an **Exchange Contract Diagram (BPMN)**.

To create an Exchange Contract Diagram (BPMN) from an interaction:

1. Right-click the interaction.
2. Select the associated exchange contract and, in its pop-up menu, click **New > Exchange contract diagram (BPMN)**.
The diagram opens with the exchange contract frame and the two *roles* representing consumer and the supplier.

 A role is a participant in an interaction, workflow or process. It can be the initiator, that is the requester of a service, or it can represent a sub-contractor carrying out processing outside the service. A role is an integral part of the object that it describes, and is not reusable. It can subsequently be assigned to an org-unit internal or external to the organization or to an IT component. Examples: client, traveler.


The *events*, *gateways* and *sequence flows* of your diagram follow the BPMN standard.


 For more details on events, gateways and sequence flows, see ["Managing events, gateways and sequence flows", page 175](#)

Defining an Exchange or an Exchange Contract Use


In an Exchange Contract Diagram (BPMN), operations are described by:

- *exchange contract use*
- *exchange use*

 An exchange contract use is associated with an exchange contract. It enables representation of complex exchanges.

 An exchange use represents the usage of an exchange in another exchange contract.

To create an *exchange contract use*:

1. Click the **Exchange Contract Use** button  and click in the diagram within the exchange contract frame.
The creation dialog box opens.
2. Click the arrow to the right of the **Specification of an Exchange Contract Use** box.

3. Select **Connect Exchange Contract** from the drop-down list and choose the exchange contract that you want to use.
4. In the **From** field, select the described exchange contract role connected to the "Consumer" role of the exchange contract use.
5. In the **To** field, select the described exchange contract role connected to the "Supplier" role of the exchange contract used.
6. Click **Finish**.

HOPEX IT ARCHITECTURE REPORTS



HOPEX IT Architecture provides facilities for analyzing and tracking the changes implemented in the IT Infrastructure of your architecture. **HOPEX** Suite uses reports to group sets of repository objects and study their interactions.

☛ *For more details on operation of reports, see the HOPEX Common Features guide, "Generating Reports".*

Report templates proposed as standard by **HOPEX IT Architecture** offer various analysis presentation possibilities. Some reports are shared with other solutions, for example **HOPEX Business Architecture**.

Several types of reports are available on several types of object.

- ✓ Exploded diagram report

☛ *For more details, see "Launching the exploded diagram report" in the diagram chapter of **HOPEX Common Features** guide.*

- ✓ **Building Block Breakdown report**

☛ *For more details, see "[Building Block Breakdown report](#)", page 186.*

BUSINESS CAPABILITIES REPORTS

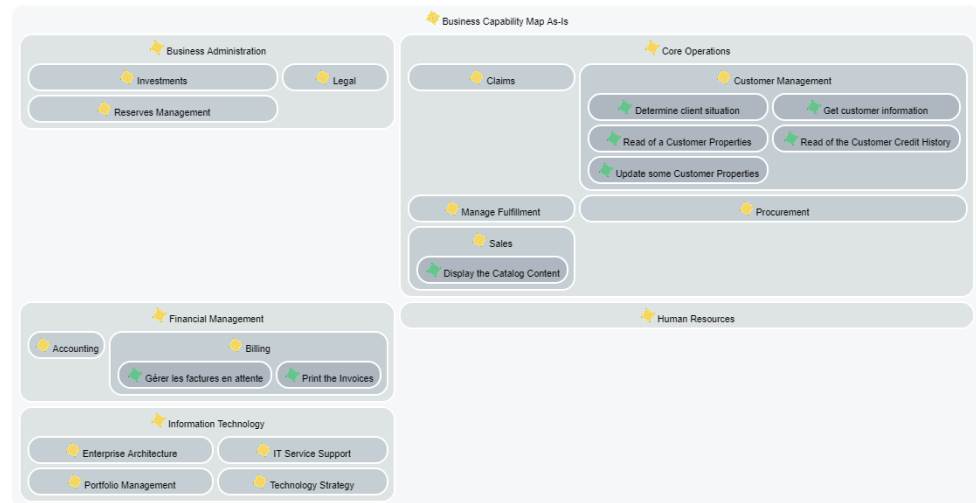
Breakdown map of business capabilities

You can use this report to display the realization coverage of business capability elements by operational elements such as logical and physical applications, application systems, etc.

Report examples

The example below enables viewing of the coverage rate of the capability map specified as parameters.

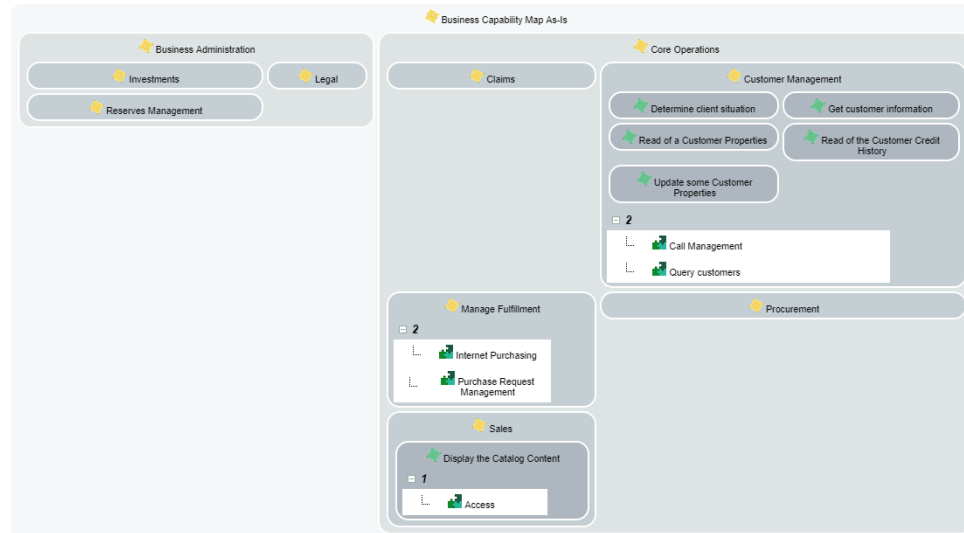
1. Business Capability Map Breakdown



For more details on how to associate a business capability with a functionality, see ["Defining the functionalities associated with business capabilities"](#), page 65.

The example below shows how the functionalities associated with capabilities are implemented by application components.

1. Business Capability Map Breakdown



For more details on how to associate a business capability with an application, see ["Creating a business capability realization"](#), page 65.

Report parameters

This consists of defining report input data.

| Parameters | Parameter type | Constraints |
|-------------------|----------------------------|--|
| Root object | Capability map, Capability | One object mandatory. |
| Depth level | Short | Defines the breakdown level of the business capability map or the capability entered as a parameter. |
| Number of columns | Short | Defines the number of columns displayed by breakdown level (for eg. 2 or 3) |

| Parameters | Parameter type | Constraints |
|---------------|---|---|
| Color palette | HOPEX palette | Mandatory. The palette delivered by default is "BoxInBox Report Monochrome Grey" |
| EA Level | Multiple choice: - business function level, - organizational level, - application level, - technical level. | Define which objects of which type of architecture level are displayed for capability realizations; <i>For example, activation of the "applications level" displays the business capability realizations for the Application System Environment, the Application Systems or the Applications</i> |
| EA dimension | Multiple choice: - capability models, - agent models, - process model, - information models, - performance models, - results models | Define which types of objects are examined within the framework of the breakdown analysis <i>For example, activation of "capability models" will display the business skills or functionalities required by the capabilities that are broken down</i> |

Matrix of Business Capabilities and Expected Functionalities

You can use this report DataSet to view links between functionalities and business capabilities.

Report examples

The example below enables viewing, in the form a table, of the coverage rate of the capability specified as parameters.

Matrix

Bar Chart

| | | |
|-------------------------|-----------------|--------------------|
| | Manage invoices | Print the Invoices |
| Accounting | | |
| Billing | ✓ | ✓ |
| Enterprise Architecture | | |
| IT Service Support | | |
| Portfolio Management | | |
| Technology Strategy | | |

For more details on how to associate a business capability with a functionality, see ["Defining the functionalities associated with business capabilities"](#), page 65.

REPORTS ON THE ARCHITECTURE FUNCTIONAL COVERAGE

Building Block Breakdown report

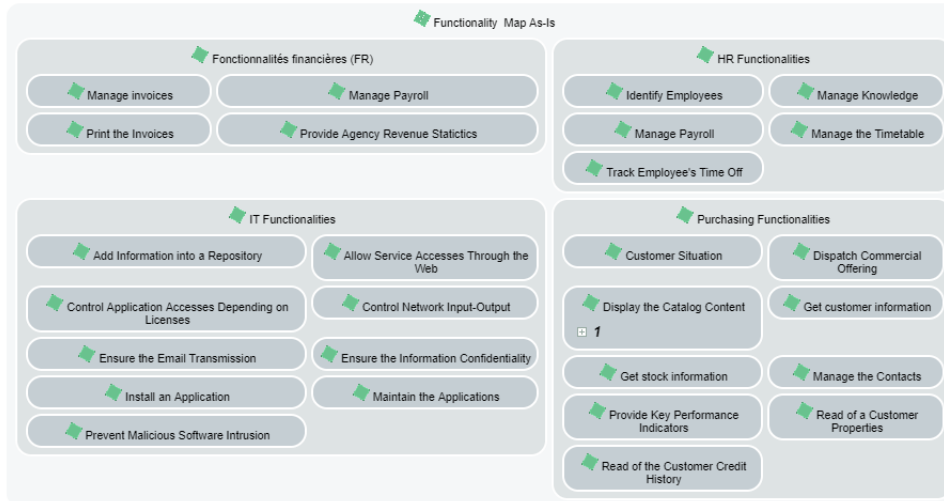
You can use this report to display the realization coverage of business capability elements by operational elements such as logical and physical applications, application systems, etc.

☛ For more details on how to associate a business capability with an application, see ["Creating a business capability realization", page 65](#).

Report examples

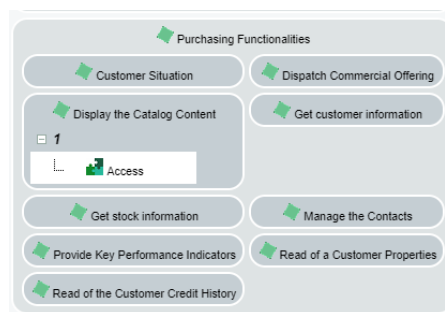
The example below enables viewing of the functional breakdown of the functionality map specified as parameters.

1. Building Block Breakdown Report



Example of functionality breakdown report.

In the example below, the applications that implement the functionalities are presented.



Report parameters

This consists of defining report input data.

| Parameters | Parameter type | Constraints |
|-------------------|---|---|
| Root object | Building block | One object mandatory. |
| Depth level | Short | Defines the breakdown level of the business capability map or the capability entered as a parameter. |
| Number of columns | Short | Defines the number of columns displayed by breakdown level (for eg. 2 or 3) |
| Color palette | HOPEX palette | Mandatory. The palette delivered by default is "BoxInBox Report Monochrome Grey" |
| EA Level | Multiple choice: - business function level, - organizational level, - application level, - technical level. | Define which objects of which type of architecture level are displayed for capability realizations; <i>For example, activation of the "applications level" displays the business capability realizations for the Application System Environment, the Application Systems or the Applications</i> |
| EA dimension | Multiple choice: - capability models, - agent models, - process model, - information models, - performance models, - results models | Define which types of objects are examined within the framework of the breakdown analysis <i>For example, activation of "capability models" will display the business skills or functionalities required by the capabilities that are broken down</i> |

Matrix of Business Capabilities and Expected Functionalities

You can use this report DataSet to view links between functionalities and business capabilities.

Report examples

The example below enables viewing, in the form a table, of the coverage rate of the capability specified as parameters.

| Matrix Bar Chart | | | |
|-------------------------|--|-----------------|--------------------|
| | | Manage Invoices | Print the Invoices |
| Accounting | | | |
| Billing | | ✓ | ✓ |
| Enterprise Architecture | | | |
| IT Service Support | | | |
| Portfolio Management | | | |
| Technology Strategy | | | |

For more details on how to associate a business capability with a functionality, see ["Defining the functionalities associated with business capabilities"](#), page 65.

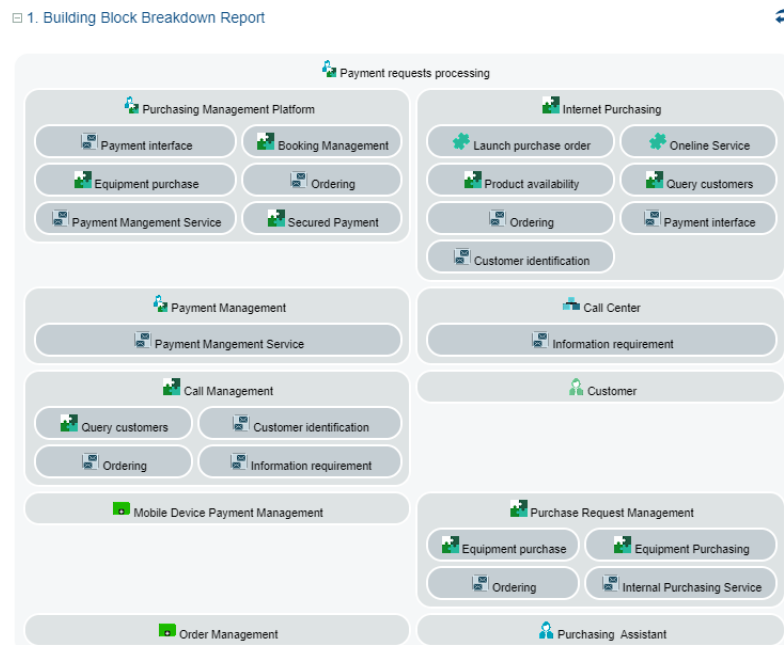
APPLICATION ARCHITECTURE REPORTS

Application Architecture Breakdown Report

Based on the same principle as the breakdown reports, this report presents the breakdown of a logical application architecture with respect to its components.

Report example

The example below shows the breakdown of the application system area for purchasing requests.



Example of a application system breakdown report

Report parameters

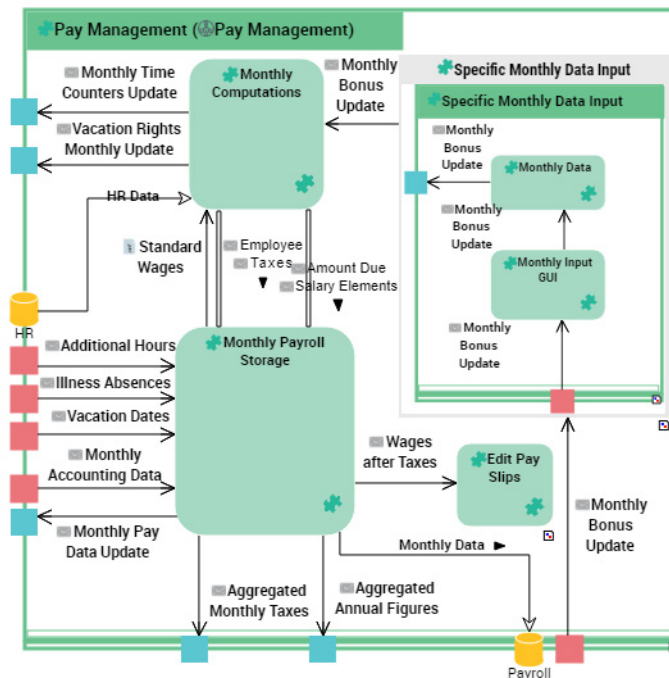
This consists of defining report input data.

| Parameters | Parameter type | Constraints |
|-------------------|---|--|
| Root object | application architecture building block | One object mandatory. |
| Depth level | Short | Defines the breakdown level of the business capability map or the capability entered as a parameter. |
| Number of columns | Short | Defines the number of columns displayed by breakdown level (for eg. 2 or 3) |
| Color palette | HOPEX palette | Mandatory. The palette delivered by default is "BoxInBox Report Monochrome Grey" |

Impact report (Scenario)

This report is used to analyze the impact of the failure of an application component on the application systems that use it or with which it exchanges application flows in the **use scenario** contexts of this application component.

For example, the "Specific Monthly Data Input" IT Service used in the context of the scenario of flow diagram below.



The report presents the list of application flows exchanged with the flow scenario components presented below.

1. Impact (Scenario)

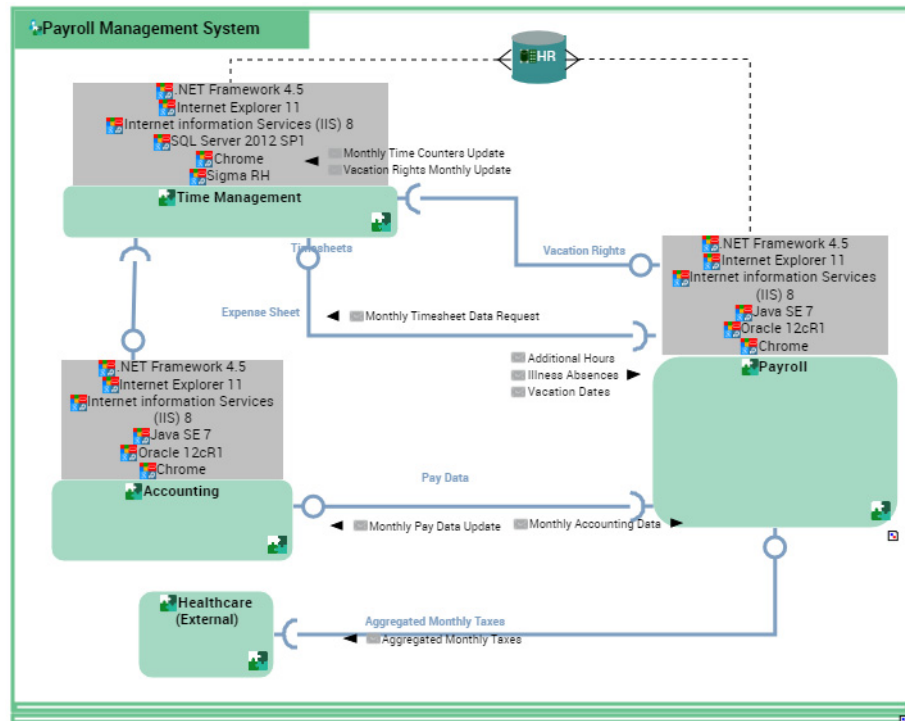
Specific Monthly Data Input

| | |
|-----------------------------|---|
| Specific Monthly Data Input | |
| Payroll | Specific Monthly Data Input Bonus in Pay Slips Management Bonus in Pay Slips Management |
| Monthly Computations | Monthly Bonus Update Monthly Bonus Update Monthly Computations Pay Management |
| Payroll | Monthly Computations Bonus in Pay Slips Management Bonus in Pay Slips Management |
| Manager | |
| Payroll Management System | Manager Bonus in Pay Slips Management Bonus in Pay Slips Management |

Impact report (Structure)

This report is used to analyze the impact of the failure of an application component on the application components that use it or with which it interacts using exchange contracts as described in its **structure diagrams** that use this application system.

For example, the "Time Management" application is used in the context of the structure diagram below.



The report presents the list of components with which the application interacts using the exchange contracts.

1. Impact (Structure)

Time Management

| | |
|---|--|
| <div> Time Management </div> | |
| <div> Payroll Management System </div> | <div> Time Management </div> |
| <div> Payroll Management System Environment </div> | <div> Payroll Management System </div> |
| <div> Payroll </div> | <div> Timesheets </div> <div> Timesheets </div> <div> Payroll Management System </div> <div> </div> <div> </div> <div> </div> |
| <div> Payroll Management System </div> | <div> Payroll </div> |
| <div> Payroll Management System Environment </div> | <div> Payroll Management System </div> |
| <div> Accounting </div> | <div> Expense Sheet </div> <div> Expense Sheet </div> <div> Payroll Management System </div> <div> </div> |
| <div> Payroll Management System </div> | <div> Accounting </div> |
| <div> Payroll Management System Environment </div> | <div> Payroll Management System </div> |

INFRASTRUCTURE REPORTS

Infrastructure Description Report

This report makes it possible to analyze the components of the infrastructure and the expected relationships between them.

➡ For more details, see ["Describing an IT infrastructure", page 117](#).

The different chapters of this report verify that each element described in an infrastructure assembly structure diagram hosts a software or hardware component.

The chapters of this report are devoted to:

- the list of networks,
- the list of computing devices,
- the list of IT technical devices,
- the list of communication channels.