



**United Nations
University**

UNU-IIST
**International Institute for
Software Technology**

Domain Models and Enterprise Application Framework for Developing Electronic Public Services

Adegboyega Ojo, Tomasz Janowski and Elsa Estevez

April 2007

UNU-IIST and UNU-IIST Reports

UNU-IIST (United Nations University International Institute for Software Technology) is a Research and Training Centre of the United Nations University (UNU). It is based in Macao, and was founded in 1991. It started operations in July 1992. UNU-IIST is jointly funded by the government of Macao and the governments of the People's Republic of China and Portugal through a contribution to the UNU Endowment Fund. As well as providing two thirds of the endowment fund, the Macao authorities also supply UNU-IIST with its office premises and furniture and subsidise fellow accommodation.

The mission of UNU-IIST is to assist developing countries in the application and development of software technology.

UNU-IIST contributes through its programmatic activities:

1. Advanced development projects, in which software techniques supported by tools are applied,
2. Research projects, in which new techniques for software development are investigated,
3. Curriculum development projects, in which courses of software technology for universities in developing countries are developed,
4. University development projects, which complement the curriculum development projects by aiming to strengthen all aspects of computer science teaching in universities in developing countries,
5. Schools and Courses, which typically teach advanced software development techniques,
6. Events, in which conferences and workshops are organised or supported by UNU-IIST, and
7. Dissemination, in which UNU-IIST regularly distributes to developing countries information on international progress of software technology.

Fellows, who are young scientists and engineers from developing countries, are invited to actively participate in all these projects. By doing the projects they are trained.

At present, the technical focus of UNU-IIST is on formal methods for software development. UNU-IIST is an internationally recognised center in the area of formal methods. However, no software technique is universally applicable. We are prepared to choose complementary techniques for our projects, if necessary.

UNU-IIST produces a report series. Reports are either Research \mathcal{R} , Technical \mathcal{T} , Compendia \mathcal{C} or Administrative \mathcal{A} . They are records of UNU-IIST activities and research and development achievements. Many of the reports are also published in conference proceedings and journals.

Please write to UNU-IIST at P.O. Box 3058, Macao or visit UNU-IIST's home page: <http://www.iist.unu.edu>, if you would like to know more about UNU-IIST and its report series.

G. M. Reed, Director



**United Nations
University**

UNU-IIST
**International Institute for
Software Technology**

P. O. Box 3058
Macau

Domain Models and Enterprise Application Framework for Developing Electronic Public Services

Adegboyega Ojo, Tomasz Janowski and Elsa Estevez

Abstract

This paper presents generic domain models to underpin the development of Electronic Public Services (EPS) – from conceptual models, through requirements and architecture, to implementation models. The conceptual model follows the analysis of 25 concrete business licensing and 6 social welfare services delivered by governments to businesses and citizens respectively. Based on this model, we characterize generic business licensing and social welfare services and, following the Governance Enterprise Architecture, synthesize a generic process for delivering Authorization and Certification classes of public services. From the generic process, requirements are obtained and the architecture is defined to support these requirements. The architecture comprises three categories of components – Front-Office, Mid-Office and Back-Office. We present the static and behavioral view of this architecture and show how it supports the variability in the development of concrete e-Licensing or e-Welfare EPS through: concrete process specification at the Mid-Office, binding of specialized tasks to automation support at the Back-Office, and use of configuration files. Finally, we discuss an Enterprise Application Framework as an implementation of the architecture based on open standards, and describe the use of the framework for rapid development of EPS based on concrete project experience. This work was carried out in the context of the e-Macao Project.

Adegboyega Ojo is a Research Fellow at the Center for Electronic Governance - United Nations University - International Institute for Software Technology (UNU-IIST-EGOV) in Macao, on leave from the Department of Computer Sciences, University of Lagos, Nigeria where he is a Senior Lecturer. Before his affiliation with UNU-IIST, he led major software development projects involving large multinationals in Nigeria. He has been involved in e-government research and development activities since 2004 at UNU-IIST as a member of the project team working on the e-government development in Macao. His research interests include Software Infrastructure for Electronic Government, E-Government Measurement, Ontology, Semantic Interoperability, and Applications of Computational Intelligence. He holds a doctorate degree in Computer Science from the University of Lagos in Nigeria. He is also a member of the Computer Professionals of Nigeria.

Tomasz Janowski is the Head of the Center for Electronic Governance - United Nations University - International Institute for Software Technology in Macao (UNU-IIST-EGOV) and a Senior Research Fellow of UNU-IIST. Tomasz Janowski is the author or co-author of numerous publications in Computer Science, Software Engineering and Foundations. He has been a Program Committee member at many international conferences, supervised a number of international students and staff, and led several research, development and capacity-building projects. These include the e-Macao Program to build a foundation for Electronic Government in Macao and the UNeGov.net initiative to building a global community of practice for Electronic Governance. As part of the UNeGov.net initiative, he organized many workshops and schools on Electronic Governance in developing countries. His current research interests include foundations of Electronic Governance, tools and applications of formal methods, and rigorous development of enterprise software, particularly software for the public sector. Tomasz Janowski holds a PhD in Computer Science from the University of Warwick, England, and an MSc in Mathematics from the University of Gdansk, Poland.

Elsa Estevez is a Project Staff at the Center for Electronic Governance - United Nations University - International Institute for Software Technology in Macao (UNU-IIST-EGOV). She is also an Assistant Professor at the Universidad Nacional del Sur (UNS) in Bahia Blanca, Argentina where she teaches subjects related to Software Engineering. Prior to joining UNS, she has been working as a Project Leader, Project Manager, System Analyst and Programmer in Information Technology for major industries, including banking, in Argentina. While in UNU-IIST-EGOV, Elsa Estevez contributed to various tasks of the e-Macao and UNeGov.net projects, from research and development, through training, to networking. Her current research interest include: Electronic Government, developing software infrastructure for Electronic Government, and Component-Based Software Development. Elsa Estevez holds an MSc degree in Computer Science from UNS and a Licenciado Degree in Computer Science from Universidad de Buenos Aires, Argentina.

Contents

1	Introduction	1
2	Conceptual Framework and Domain Analysis	2
2.1	Typologies of Public Services	2
2.2	Business Licensing Services	4
2.3	Social Welfare Services	6
2.4	Public Service Delivery - Conceptual Model.....	7
2.5	Common Features in Public Service Delivery	8
3	EPS Domain Requirements	8
4	Generic EPS Architecture	9
4.1	Architectural Elements	9
4.2	Behavioral Model.....	11
4.3	Variability Support in EPS Architecture	12
5	EPS Application Framework	12
5.1	Implementation Technologies	13
5.2	EPS Framework-Based Application Engineering	13
6	Conclusions	14
	References	14

1 Introduction

The provision of Electronic Public Services (EPS) is central to measuring e-government efforts in terms of number, maturity and availability of these services to all key stakeholders – citizens, businesses and government itself. According to the last measurement of online maturity of twenty basic public services carried out within the European Commission’s survey framework [5], about 50% of these services are now fully available online – reached the highest level of maturity – across the EU Member States. However, online delivery of EPS is still posing significant challenges in other developing regions of the world for various reasons, from organizational preparedness, to cost of development and sustainability.

Large-scale development and deployment of EPS requires significant investment in software infrastructure. Such infrastructure must support the rapid development and deployment of EPS. Most software infrastructures in use provide deployed EPS with basic features and reusable business patterns to address the issues of: security and authentication, user profiling, e-payment, auditing and notification, address verification, acknowledgement, etc. For instance, the Public Service Infrastructure Initiative (PSi) of the Government of Singapore [9][10] is unique among similar efforts in supporting the development, testing and deployment of EPS, in addition to providing basic application services. However, domain models underpinning the development of the PSi infrastructure as well as development of infrastructure-supported EPS are not available in the public domain for reuse.

The availability of domain models for EPS development and their implementation as reusable, open-standards-compliant components within an EPS Infrastructure are critical to large-scale development and deployment of EPS by governments. Our observation so far shows limited results in this direction. For instance, no open source implementation of a single EPS has been posted on the IDABC (Interoperable Delivery of e-Government Services to Administration, Business and Citizens) Open Source Software Repository page (<http://ec.europa.eu/idabc/en/chapter/5649>) for the past two years despite the number and sophistication of electronic services across Europe.

This observation has partly motivated the work reported in this paper. Another motivation comes from the authors’ concrete experience in developing e-Government services and infrastructure as part of the e-Macao Project. The experience highlights a clear need to base EPS and infrastructure development on generic domain models as part of a clearly-defined Enterprise Application Framework.

The paper presents our approach to the development of an Enterprise Application Framework for the development of EPS. Our methodology follows the basic steps of a typical domain engineering process, from domain analysis, through domain requirements, to domain design and implementation [11]:

- 1) Domain Analysis – This is carried out through the analysis of 25 business licensing services and 6 social welfare services offered by Government to businesses and citizens in Macao [12] and a study of the Governance Enterprise Architecture (GEA) Object Model [14].
- 2) Domain Requirements - A generic set of requirements for a typical Electronic Public Service are obtained from feature analysis of concrete requirements posed by Social Welfare Benefits and Business License Issue services against ESP supporting them, and validated based on the GEA Model.

- 3) **Domain Design** - This consists of a generic architecture for a typical Electronic Public Service and the specifications of behavioral aspects of this architecture. The architecture consists of Front-Office (FO), Back-Office (BO) and Mid-Office (MO) subsystems. The FO subsystem supports the specification and submission of requests by users. The MO subsystem allows for the mapping of user requests to concrete business processes through service composition and orchestration or through the enactment of workflows. The BO subsystem consists of a task manager and bindings to a suite of applications for implementing the various steps of the business processes enacted at the MO.
- 4) **Domain Implementation** - An Enterprise Application Framework based on open standards and open-source software such as J2EE, XML and Web Services provides a prototype implementation of the design. Based on this Enterprise Application Framework, a development process is described to carry out concrete EPS development through specialization and extension.

The rest of the paper is organized as follows. Section 2 describes the various types of government services with emphasis on business licensing (e-Licensing) and social welfare (e-Welfare) services, and provides a conceptual model for the domain of EPS. Section 3 shows how generic domain requirements for any Electronic Public Service are obtained from concrete requirements for e-Licensing and e-Welfare systems. A generic component design for an EPS is presented in Section 4, while Section 5 discusses how this design can be implemented. Finally, some conclusions are outlined in Section 6. Conceptual Framework and Domain Analysis

2 Conceptual Framework and Domain Analysis

We describe in this section the theoretical framework underpinning our work and also provide a conceptual model resulting from the domain analysis of public service delivery. Our approach to domain analysis is three-fold: (i) theoretical - conceptual characterization of public services based on the Governance Enterprise Architecture (GEA), (ii) empirical - analysis of 25 business licensing services and 6 social welfare services and (iii) mixed - relating both approaches. As discussed later, this approach yields a cross-validation of the GEA model and our findings.

2.1 Typologies of Public Services

Governments are offering many services to their customers and stakeholders. A common way of characterizing these services is based on their recipients - citizens, businesses, government, visitors, etc. For instance, the 20 basic public services adopted for electronic delivery by the EU are categorized into two basic classes: services for citizens and services for businesses [5]. These twenty services are also classified for analytical purposes into four service clusters: Income Generating Services, Return Services, Registration Services, and Permits and License Services. In this clustering, services related to income tax, value added tax, corporate tax, customs declaration and social contributions are under Income Generating Services. A sector-oriented classification is provided by the Business Reference Model of the Federal Enterprise Architecture (BRM-FEA) [6]. This model classifies citizen services into 19 classes including: Community and Social Services, Defense and National Security Services, Disaster Management Services, Economic Development Services, Education Services, Energy Services, etc. Yet another classification is offered in [14][15] as part of the Governance Enterprise Architecture (GEA), in which four types of public services are identified: Certification, Authorization, Control and Production.

Both BRM-FEA and GEA provide information on the mode of delivery of these services or more accurately, the nature of the Public Administration (PA) function that is associated with these services. For instance, BRM-FEA defines some business areas including - knowledge creation and management, public good creation and management, and regulatory compliances and enforcement [6], while GEA [14] defines three abstract types of administrative functions - declarative, direction and interrogative, by modeling the interaction between administration and citizens based on linguistic theory.

The GEA typology appeals to us for a number of reasons. First, it provides clear operational definitions of administrative functions and public service types (see

Table 1 and

Table 2, following [14]), as against simple enumeration in other schemes. Second, typical steps of three of the four types of public services are specified. For instance, both licensing and social welfare services may be viewed as Authorization services delivered administratively in the permissive and supportive modes respectively. This logically implies some degree of commonality between these two categories of services. The commonalities and differences in a system to deliver these two kinds of services are explored next.

Function	Description	Pattern
declarative	PA declares and certifies the existence and truth of certain world states.	certifying "X" for entity "Y"
imperative-permissive	PA directs the society to a given state through commands. In the imperative mode, PA mandates some societal behavior. In the permissive mode, PA recognizes some special rights and allows behaviors otherwise prohibited.	do "X" (imperative) "X" is prohibited unless "Y" (permissive)
supportive	PA directs the society to a given state through supportive actions.	grant "X" to entity "Y"
interrogative	PA interrogates to collect societal needs as a feedback mechanism for declarative and directive actions.	ask "X" about "Y"

Table 1: GEA Administrative Functions

Service Type	Description
certification	PA declares and certifies different states of the world through the declarative function, e.g. issuance of personal documents.
control	PA fulfils the imperative function by ensuring compliance, e.g. inspection services.
authorization	PA together realizes permissive and supportive functions through this category, e.g. licensing services.
production	PA organizes production mechanism, e.g. utility services.

Table 2: GEA Public Service Types

For instance, following [14], generic process descriptions to deliver certification and authorization services are depicted in Table 3 and

Table 4 below. Based on the GEA characterization of services and descriptions of different process steps, we make a number of observations expressed as propositions below:

- 1) Certification–Authority Similarity - The processes associated with the Certification and Authorization services are very similar and could be easily streamlined into a single process.
- 2) Authorization–Certification Concomitance - Certification services are inherently part of an Authorization services, since evidences provided would have to be verified.
- 3) Control as Authorization Support - Control services provide a basic mechanism for monitoring adherence to the rules governing a granted authorization.

An entity applies for a certification.
 PA asks for evidences needed for certification.
 An entity submits necessary evidences to PA.
 PA checks completeness and correctness of all evidences.
 PA processes all inputs based on established rules, checking if all prerequisites are satisfied.
 PA asks for additional evidence and clarifications if required.
 PA certifies or justifies the refusal.

Table 3: Certification Process

An entity asks for authorization - permission or support.
 PA asks for specific evidences to be provided.
 An entity submits necessary evidences to PA.
 PA checks completeness and correctness of all evidences.
 PA processes all inputs received, checking if all prerequisites for authorization are satisfied.
 PA requests additional evidences/clarifications if required.
 PA certifies or justifies the refusal.

Table 4: Authorization Process

Possible classification of Business Licensing and Social Welfare as Authorization services provides some grounds for abstraction of these services into one. In addition, the possibility of streamlining process steps for Authorization and Certification into a single process shows that these two types of services could be treated as a single category and consequently provided the same support. It is therefore plausible to claim that our choice of Business Licensing and Social Welfare services is representative of Authorization and Certification services and provides a starting point in the development of a domain model for EPS.

2.2 Business Licensing Services

Licensing services entail granting various kinds of government permissions to citizens, businesses, visitors, associations, etc. There are at least three types of permissions: (i) Licensure – right to practice, (ii) Certification – right to title, and (iii) Permit - right to perform some regulated tasks. From a survey of government agencies conducted by the authors in 2004, licensing services feature prominently among the several services provided by government, especially to businesses [12]. A representative list is depicted in

Table 5. License-related services include: application for a new license, renewal of an expired license, re-instating a revoked license, tracking the progress of a license application, and booking an appointment with a licensing agency for information on how to submit an application.

No	Licensing Service	No	Licensing Service
1	temporary/permanent electricity license	14	radio network license
2	construction and utilization license	15	radio station license
3	aviation industry license	16	nursing home establishment license
4	certificates of origin license	17	pharmacy license
5	import and export license	18	private educational institution license
6	trademark registration	19	adult education and tutorial centre license
7	temporary and permanent factory license	20	tourist guide
8	auditing firm license	21	travel agency license
9	media house registration	22	bank license
10	marine operations and works license	23	money exchange agent license
11	marine taxi license	24	remittance company license
12	food and animal origin license	25	financial intermediaries license
13	food and beverage license		

Table 5 : Business Licensing Services

The process depicted in Figure 1 is a simplified workflow for the 25 licensing services in Table 5. A detailed description of different process steps is provided in [1].

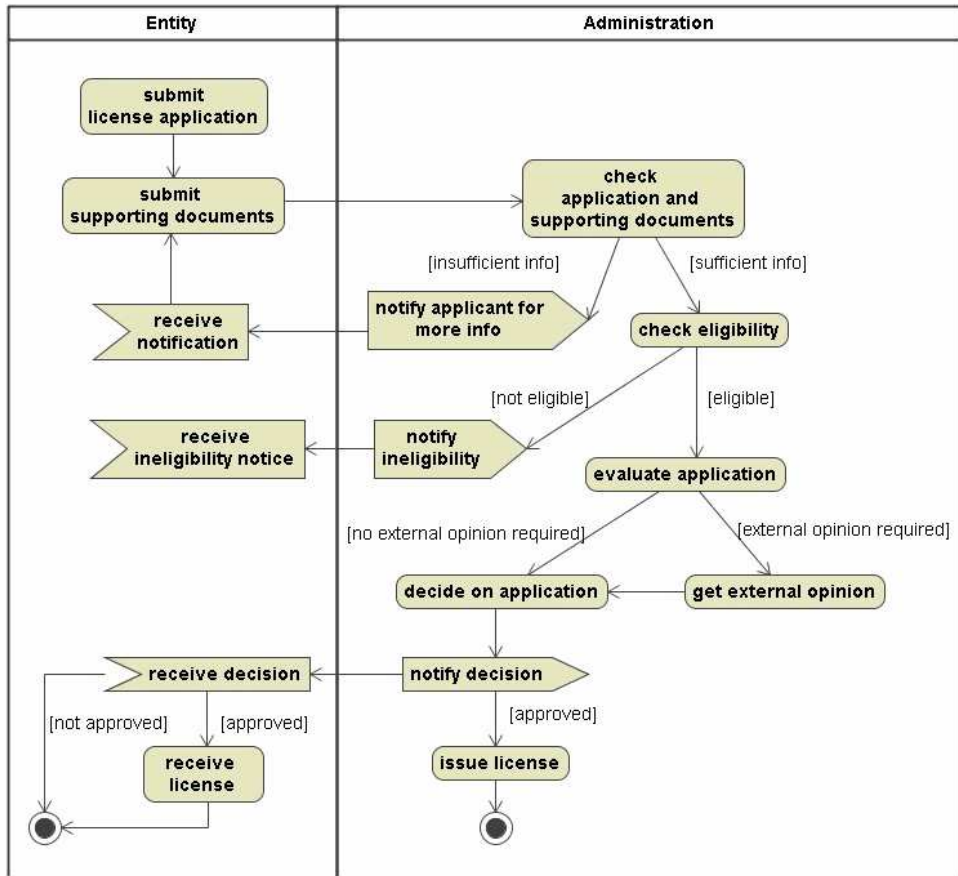


Figure 1: Workflow for Licensing Services

2.3 Social Welfare Services

Through these services, governments assist different categories of citizens, such as: provide financial assistance to families and individuals with low incomes; provide allowances on occasions of marriage or child birth; grant post-graduate scholarships to students, etc. To receive these benefits, citizens are required to meet some requirements related to: minimum age, residency, civil status and others. From the survey of government agencies carried out by the authors in 2004, a set of Social Welfare services were identified [12][8]. Six of them were selected for the purpose of carrying out domain analysis in this paper.

No	Welfare Service	No	Welfare Service
1	social houses	4	financial assistance to individuals and families
2	financial aids to students	5	post-graduate scholarships
3	retirement pensions	6	survivor pensions

Table 6: Social Welfare Services

The nine step process described in Table 7 below is required for delivering a typical Social Welfare service. Details of this process are fully documented in [8].

A citizen submits an application for a benefit.
 PA optionally requests supporting documents or evidences.
 The citizen submits the requested supporting documents.
 PA verifies the submitted application and supporting documents.
 PA decides on the applicable benefit.
 PA optionally requests additional information or clarifications.
 PA approves or declines the application.
 PA notifies the citizen.
 The citizen receives the benefit, if approved.

Table 7: Social Welfare Process

2.4 Public Service Delivery - Conceptual Model

We present a conceptual model for the domain of public services. The model identifies and relates a set of general concepts that define this domain. By ensuring that the key aspects of public service delivery are captured, the model provides the semantic basis for formulating the domain requirements discussed in subsequent sections. The model is based on the Government Common Information Model [4].

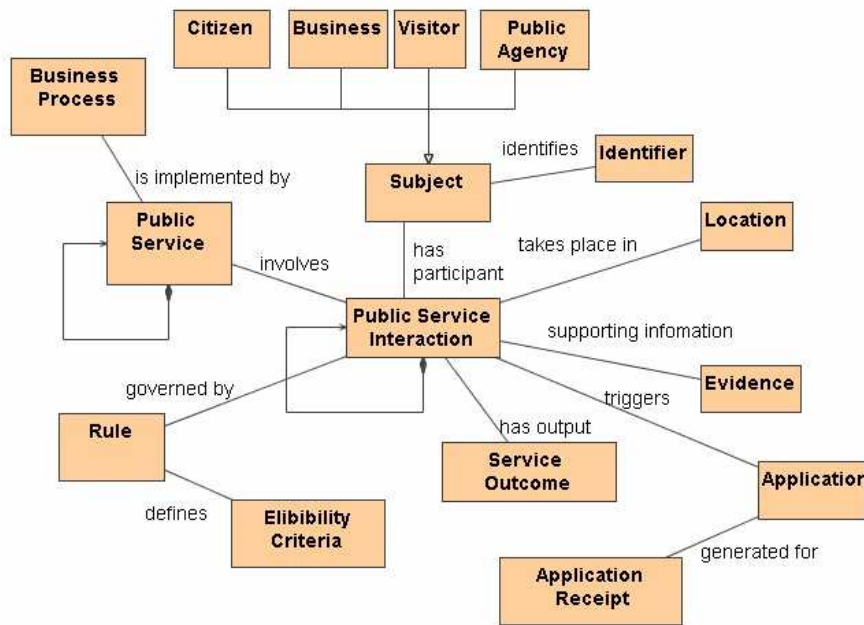


Figure 2: Public Service Delivery - Conceptual Model

The Public Service Interaction concept is central to the model, typically describing some exchange of information among defined participants (Subjects) to enable the delivery of a Public Service. These participants typically include a Public Agency that provides the service and an entity requesting the service, such as a Citizen, a Business, a Visitor, etc. The participants may also include third-party government agencies as is the case of licensing services where opinions on applications may be sought externally. Each participant in the interaction can be unambiguously identified through an Identifier. For instance, a Citizen will have an identity card number and a Visitor will have a passport number. Service Interaction occurs at some Location, whether online or physically in a government office. Evidences or supporting documents are for processing Applications. The Eligibility Criteria are determined for an applicant by the Rules governing the service interaction. Each submitted application is provided an Application Receipt by the PA. Public Services are implemented at the PA by well defined Business Processes. A Public Service Interaction may generate or involve several service interactions. For instance, while checking for the validity of evidence, certification services may be triggered either within the same public agency or in a third-part agency. A Public Service may also be a composite of various services. Finally, all service interactions must result in an outcome which is called Service Outcome.

2.5 Common Features in Public Service Delivery

From the conceptual model depicted in Figure 2 and the sequence of activities given in Figure 1 and Table 7, the basic steps in delivering public services can be described as follows: (i) submission of application by an applicant, (ii) submission of supporting documents by an applicant, (iii) verification of the application and supporting documents for correctness and completeness, (iv) notification of the applicant either to provide more information or to inform about the outcome, (v) making a decision on the application, and (vi) issuing or granting the requested support (benefit) or authorization (license).

These steps are clearly similar to the steps of the Certification (Table 3) and Authorization Processes (Table 4) of the GEA model respectively. This similarity allows us to: (i) validate the outputs of our domain analysis based on the studies of concrete public services, (ii) validate some aspects of the GEA model and (iii) define a minimum generic process of a typical certification or authorization type public service. This process provides the basis for capturing domain requirements in the next section.

An entity submits an application for a service.
 PA optionally requests supporting documents or evidences.
 The entity submits the necessary supporting documents.
 PA verifies the submitted application and supporting documents.
 PA optionally requests additional information or clarifications.
 PA decides on the request.
 PA approves or declines the request.
 PA notifies the entity about its decision.
 The entity is granted the request, if approved.

Table 8: Generic Process for Public Service Delivery

3 EPS Domain Requirements

Public services can be delivered at different levels of maturity: from information on the procedure how to apply for a service, through online support to downloading and uploading application forms and supporting documents, to fully transactional services. The EPS Application Framework supports fully transactional services. We only discuss core business requirements. Information on basic

technical requirements such as user management, logging of transactions, and authentication and security are documented in [3][7].

Table 9 presents high-level use cases for the EPS Application Framework. Refinement of these use cases and detailed specification of the associated informal requirements can be found in [3][7]. These use cases can be easily mapped to the generic process displayed in

Table 8. The interaction between the Entity and the Public Agency in obtaining a public service is initiated by the Entity through the Submit Application use case. Next, the Entity uploads specified supporting documents through the Upload Supporting Documents use case. Once the application is submitted and the application receipt obtained, the Entity may use the receipt to determine the status of the application using the Track Application use case. Upon receiving the submitted application and supporting documents, the Public Agency first verifies completeness through the Check Validity and Completeness of Application Documents use case and proceeds to use the Verify Evidences use case to verify supporting documents. The eligibility of the Entity to receive requested service can be automatically evaluated through the Assess Eligibility use case. Decision making by the Public Agency on any application is carried out using the Decide on Application use case. All forms of entity notification, irrespective of the channel, are done through Notify Entity.

No	Use Case	Actor
U1	Submit Application	Entity
U2	Upload Supporting Documents	Entity
U3	Check Validity and Completeness Of Application Documents	Public Agency
U4	Verify Evidences	Public Agency
U5	Assess Eligibility	Public Agency
U6	Notify Entity	Public Agency
U7	Track Application	Entity
U8	Decide On Application	Public Agency

Table 9: Domain Use Cases

To realize these basic functionalities as reusable domain assets, various components are designed and organized into subsystems, guided by best practices in Enterprise Architecture research and development. This architecture is elaborated in the following section.

4 Generic EPS Architecture

This section presents a generic architecture for an Electronic Public Service. This architecture serves two basic purposes: (i) a reference architecture to guide the development of specific EPS architectures later during the application engineering phase and (ii) a basis for implementing core domain components within the context of an Enterprise Application Framework. We defer discussion of the domain architecture as a reference model to Section 5, and elaborate on its components, configuration and behavior here.

4.1 Architectural Elements

The various components implementing the requirements identified in Section 3 are distributed over three major subsystems: Front-Office, Mid-Office and Back-Office. These subsystems interact asynchronously through connector components. These major elements and connectors are briefly discussed below:

- 1) **Front-Office (FO):** The components of the FO directly support the interaction between the applicant and the system while executing the use cases: Submit Application, Upload Supporting Documents and Track Application. The components: (i) receive requests from the client tier, (ii) validate requests, (iii) generate application receipts, (iv) generate exceptions and (v) store these requests in a database for further processing. The FO exposes a web service interface to the client tier thereby supporting as many client protocols as possible - HTML, XHTML, WAP, etc. The FO subsystem also supports query-related requests, by looking up appropriate databases and transferring results to the client tier. In addition, this subsystem provides an interface for implementing the eligibility evaluation component. Common functionalities, such as transaction logging, user profile management and authentication are also supported by this subsystem. The FO comprises the Service, Business Components and Data Access layers. Detailed specification of the components in the FO subsystem is provided in [3].
- 2) **Mid-Office (MO):** Requests received by the FO are forwarded through a connector component to the MO subsystem. The MO is a workflow engine, or more generally a business process management system, which associates received requests with concrete business processes. The processes are defined and loaded into the workflow a-priori. Each received request is expected to trigger an instance of at least one known business process, for instance “Apply for New License Service”. The business process consists of a fixed number of steps. The workflow engine generates a task for each process step and assigns a role to handle the task at the BO. The generated task is transferred through some connector components to the BO from where BO roles can access the tasks through an application or service. The MO subsystem also receives signals notifying the completion of tasks from the BO subsystem and generally provides state management information on user requests to support request tracking. Detailed information on the MO subsystem is provided in [1].
- 3) **Back Office (BO):** The BO subsystem is composed of a Task In-Tray component and a collection of components and services required for carrying out processing tasks received from the MO subsystem. Specifically, the BO components support the execution of the following use cases: (i) verification of evidences, (ii) evaluation and decision, (iii) tracking of applications and (v) notification of applicants. The components also provide auxiliary support for: viewing applications, viewing supporting documents, receiving tasks from the MO, and forwarding end of task signals to the MO subsystem. Detailed information about the BO sub-system is provided in [7].
- 4) **Connector Components:** Four message queues (message-oriented middleware components) are used to connect the FO, MO and BO subsystems in the architecture. These queues are: IncomingRequest Queue - forwarding requests from FO to MO, (ii) TasksToBackOffice Queue - forwarding tasks from MO to BO, (iii) EndOfTaskSignal Queue - forwarding end-of-task signals from BO to MO, and (iv) ReplyFromBackOffice Queue - forwarding synchronous requests from BO to FO.

Figure 3 presents the static view of the Generic EPS Architecture, including the FO, MO and BO subsystems and the four message queues to connect them together: IncomingRequest Queue, TasksToBackOffice Queue, EndOfTaskSignal Queue and ReplyFromBackOffice Queue.

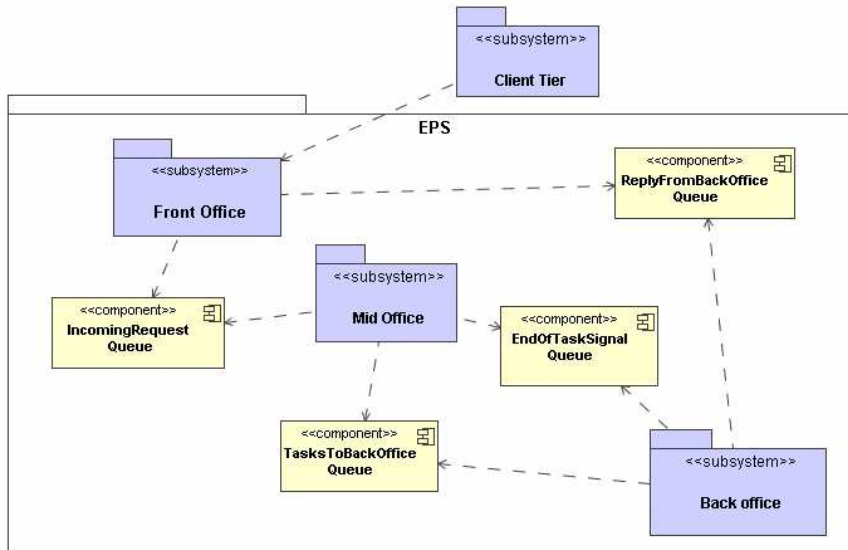


Figure 3: Generic EPS Architecture - Static View

A point to note in the presented architecture is that the major subsystems are independent of one another and only rely on these connectors for communication. In addition, despite the use of message queues to carry out asynchronous communication, the architecture also permits and supports the execution of synchronous communication required by requests such as application tracking.

4.2 Behavioral Model

The interactions between three major subsystems of the architecture are shown in the communication diagram of Figure 4. Here are two scenarios for asynchronous and synchronous requests:

- For asynchronous requests, the interaction is triggered by a request submitted by a client through the FO subsystem (1:submitRequest). The FO generates an application receipt (2:generateReceipt) once a request is submitted. The submitted request is stored in a database (3:storeRequestInDatabase) before it is dispatched to the IncomingRequest Queue (4:publishRequest). The MO retrieves the request from IncomingRequest Queue (5:getRequestFromQueue) and creates a process instance associated with the obtained request (6:createProcessInstance). Once the execution of the process instance starts (7:startProcessInstance), the current process task is assigned a role (8:assignResourceToCurrentTask) and the MO inserts this task into the TasksToBackOffice Queue (9:publishTask). The BO subsystem retrieves tasks from the task queue (10:getTask) and executes the task (11:executeTask). On completion of the task, the BO publishes the end-of-task signal in the EOTSignal Queue (12:publishEOTSignal). The MO retrieves the EOT signal from the queue (13:getEOTSignal) and transits to the next task of the executing process (14:nextTask) if there are more tasks in the process to execute.
- For synchronous requests, such as tracking requests, the BO subsystem publishes the response in the ReplyFromBackOffice Queue (15:publishResult) after publishing the EOT signal. The FO obtains the required response by retrieving the messages from the ReplyFromBackOffice Queue (16:getResult).

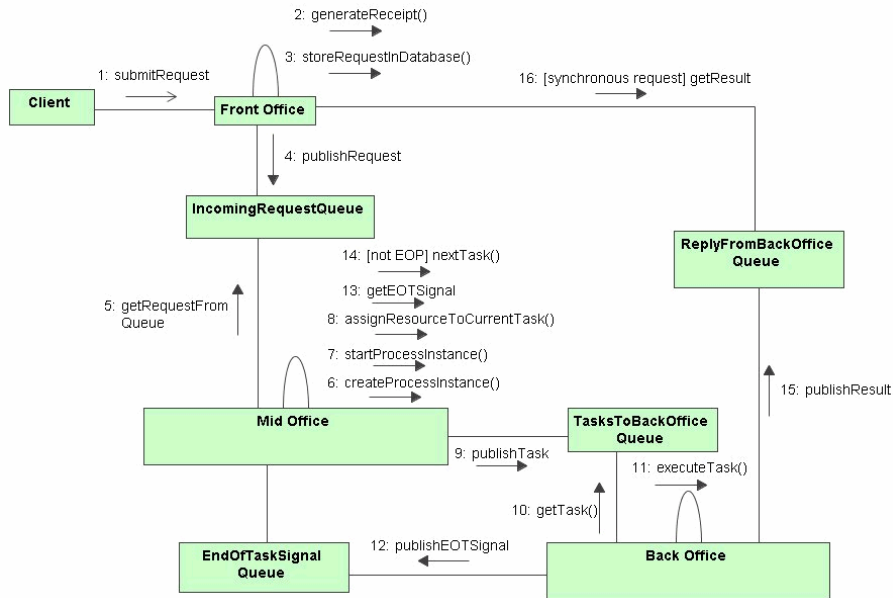


Figure 4: Generic EPS Architecture - Behavioral View

4.3 Variability Support in EPS Architecture

In this section, we describe aspects of the architecture which support variations between concrete instances of EPS. The architecture employs two related strategies to accommodate such variations: specializations and extensions. The two strategies are described as follows:

Specializations: Framework Hotspots are the points in the architecture where specialization is required and adaptation occurs. The MO subsystem provides a hotspot for defining a business process without any restriction whatsoever. Requests received by MO are mapped to unique registered business processes through the use of configuration files. Another MO hotspot allows role and resource allocation to be specified as required for generated tasks.

Extensions: Extension features allow the architect to easily add desired additional features. The service-oriented architecture of the FO allows new features to be easily added to the interface exposed to client applications in its Service layer, and to provide the necessary supporting components at the Business Component layer. In addition, while the FO operations of most governments are largely the same, the details of BO processing can vary greatly. However, since the MO subsystem allows the definition of business processes which orchestrate BO applications, BO extensions can be easily accommodated by providing more components in addition to the already available ones.

5 EPS Application Framework

This section presents the implementation of the generic EPS architecture as an Enterprise Application Framework to support the development of new EPS. Core implementation technologies are briefly discussed and the prescribed application engineering process is explained.

5.1 Implementation Technologies

The EPS Application Framework consists of two enterprise applications frameworks – the Front-Office Framework and the Back-Office Framework, and a workflow engine. The two Enterprise Application Frameworks were developed using Java/J2EE related technologies. In particular, the Spring Application Framework [13] was adopted as a solution to implement the flexibility support in the generic architecture. Spring is a J2EE-based application framework for managing the lifecycle of business components and their dependencies. This framework allows for loose coupling of classes and specification of dependencies between classes in an XML language. Web Services technology was used by the Front-Office Framework in order to receive requests from a variety of client applications capable of Web Service invocations. This potentially allows for the development of multi-channel Front-Office applications based on the framework. The JBoss Java Business Process Management System (JBPM) was employed by the MO as the workflow engine. This engine supports the definition of business processes in the native Java Process Development Language (JPDL) and the de-facto Business Process Execution Language (BPEL). XML/XML Schema is extensively used within the framework for the definition of request types and configuration documents.

The available reference implementation only considered open standards and open-source technologies. This decision guarantees the possibility of integrating systems developed on the framework with existing information systems and legacy applications. In addition, since the framework is open-source, its implementation could be modified as required or completely overhauled, if desirable. Finally, the framework can be made available as open source software.

5.2 EPS Framework-Based Application Engineering

The hotspots in the framework are the primary means for application engineering in the development of specific EPS. The implementation of the EPS architecture based on the Spring framework further facilitates extension, replacement and customization of existing framework components. The process for developing a specific Electronic Public Service based on our framework includes the following steps:

- 1) capture requirements for a new EPS, aligned with domain requirements;
- 2) develop an architecture for the new EPS using the generic EPS architecture as a reference model;
- 3) define an additional request schema or modify an existing one;
- 4) develop or customize framework components for FO Application and BP Application development;
- 5) define business processes, deploy them in MO for each request type, and bind request types to them;
- 6) modify configuration files to setup three message queues;
- 7) integrate the FO, MO and BO applications using the message queues; and
- 8) test the new Electronic Public Service over each request type.

This process was successfully adopted in the development of a prototype e-License application on top of the EPS framework. Details of the development models are documented in [3][7].

6 Conclusions

This work was motivated by the demand from e-Government practitioners to carry out large-scale development of Electronic Public Services and the underlying Software Infrastructure for e-Government upon rigorously defined domain models and application frameworks. We are not aware of the existence of such models and frameworks for representative public services in the open domain.

To develop domain models, a bottom-up domain strategy was adopted based on the e-Macao development experience [1][8], and validated top-down with the Governance Enterprise Architecture [14]. Our validation approach had a dual effect - on the one hand, we established that our generic process model was reasonable, on the other - we provided an independent validation of the GEA model. Importantly, we established that the EPS domain model and the resulting application framework cover the class of public services characterized by the Authorization and Certification types according to GEA. Our architecture implicitly supports variability like any other domain architecture. The implementation technology adopted for the application framework guarantees openness both in terms of accessibility and integration. A prototype development based on the EPS Application Framework was successful.

Future work includes the implementation of the EPS framework into a production-quality system as part of e-government infrastructure [2] and its use by agencies in developing their own EPS.

References

- [1]. Adegboyega Ojo, Gabriel Oteniya, Chau Keng Fong, Elsa Estevez, Tomasz Janowski, Electronic Delivery of Licensing Services – Development Document, e-Macao Project Report 5, October 2005.
- [2]. Adegboyega Ojo, Tomasz Janowski, Gabriel Oteniya and Elsa Estevez, Infrastructure Support for e-Government – An Overview, e-Macao Project Report 6, August 2006.
- [3]. Adegboyega Ojo, Chu Tang Jeong, Gabriel Oteniya, Tou Chi Pio and Tomasz Janowski, Front-Office Framework for e-Government, – Development Report, e-Macao Project Report 8, 2006
- [4]. Cabinet Office, Office of the e-Envoy, e-Services Development Framework Primer, February 2002.
- [5]. Capgemini, Online Availability of Public Services: How is Europe Progressing?, Web based Survey on Electronic Public Services Report of the 6th Measurement, June 2006.
- [6]. Executive Office of the President of the United States, FEA Consolidated Reference Model Document, May 2005.
- [7]. Elsa Estevez, Wan Chon, Wong Chan Tang, Adegboyega Ojo, Gabriel Oteniya and Tomasz Janowski, Back Office Framework for e-Government – Development Report, e-Macao Project Report 9, 2006
- [8]. Gabriel Oteniya, Elsa Estevez, Dai Zhen Zhong, Adegboyega Ojo, Tomasz Janowski, Electronic Delivery of Social Welfare Services – Development Document, e-Macao Report 4, October 2005.

-
- [9]. Infocomm Development Authority (IDA) – Ministry of Finance of Singapore, Factsheet of Public Services Infrastructure, September 2003.
- [10]. Jeffery BH Tan and James SL Yong, Many Agencies, One Government – Singapore’s Approach to Public Services Delivery, James SL Yong (Ed.): E-Government in Asia, 2003, pp. 204 – 232.
- [11]. Maarit Harsu, A Survey on domain engineering, Report 31, Institute of Software Systems, Tampere University of Technology, December 2002
- [12]. Tomasz Janowski, Adegboyega Ojo, Elsa Estevez, State of Electronic Government in Macao, Volume 2, e-Macao Project Report 2, April 2005
- [13]. Interface2I, Spring Framework, available online at: <http://www.springframework.org/>
- [14]. Vassilios Peristeras and Konstantinos Tarabanis, Governance Enterprise Architecture (GEA): Domain Models for E-Governance, (Ed.) Marijn Janssen, ICEC’04, 6th International Conference on E-Commerce, ACM, 2004.
- [15]. Vassilios Peristeras and Konstantinos Tarabanis, Advancing the Government Enterprise Architecture – GEA: the Service Execution Object Model, R. Traaunmuller (Ed): EGOV 2004, LNCS 3183, pp. 476-482, 2004.