DTGov: Digital Transformation of Government Business Processes

Nuno Marques
Supervisor: André Vasconcelos
Instituto Superior Técnico, Universidade de Lisboa, Portugal
{*nuno.r.marques, andre.vasconcelos*}*@tecnico.ulisboa.pt*

Keywords: Digital Transformation, Government, Processes, Low-code, Reference Architecture

Abstract: Today, companies cannot rely on outdated systems, governments being no exception. There are multiple benefits when performing the digital transformation (DT) of its processes, but also several downsides such as costs. This paper presents first the main use cases: resolve a procedure, send a notification by hand, reclassify a procedure, and create a request, and then the reference solution divided into 3 components. First, the Domain Model and Lifecycles related to the main entities. Second, the Reference Architecture adds the remaining projects' modules (Roles,...), and also the dependencies between them. Third, the Implementation presents how to implement the information in a real-world case using "low-code" technology, especially useful to mitigate several issues. Lastly, an evaluation of the proposed solution is also presented.

1 INTRODUCTION

Digital Transformation, although presenting some downsides such as interoperability problems or costs, is crucial in today's world, as seen in the biggest companies like Amazon or Alibaba (Kimberling, E., 2021). It provides many benefits such as improving the delivery of services or reducing costs in the long run. The negatives, however, cannot be overlooked., especially in governments since they deal with large quantities of personal data, creating security and privacy concerns, which can be difficult to control.

One way to ease many of the existing problems is using low-code technologies, a "collection of tools that enable the visual development of applications through modeling and a graphical interface" enabling "developers to skip hand-coding, speeding up the process of getting an application to production" (OutSystems,). Since it focuses on the feedback aspect, leaving the complications of coding, it is very useful in this type of DT, with one issue, however, that since it is relatively new, there are few studies/information regarding its implementation in real-world scenarios.

This paper aims to provide that information and how to implement it using Enterprise Information Management (EIM) systems and low-code solutions. Since, however, the work done here is based on a real-world implementation using OpenText AppWorks (OpenText,), some information is slightly more towards using it, but overall, generalized enough to be used in other low-code applications.

Since the term DT is very broad, the goal is to focus on the DT of government's processes, incentivizing governments to implement or improve it, specifically on the employees' side.

To achieve that, first it is presented the related work containing a Systematic Literature Review (SLR) that identifies the principal modules used in similar DTs. From that, and the real-world implementation used as a basis for this paper, the main use cases are identified. Following that is created the solution proposition, divided into 3 parts: the first one refers to the creation of entity models and their lifecycles, the second one presents the reference architecture created, and the last one refers to how to implement that information using low-code technologies. Lastly, there is the evaluation of the work and the conclusion, mentioning this paper's main contributions and how to improve it in future work.

2 RELATED WORK

In this section, two SLRs were performed, the first one to better understand the current representations (state of art, frameworks, and models) of DT, serving as the background and motivation for the presented work, and the second SLR more specific, such as modules present in the DT. Important to note that both follow the same steps and methodologies.

2.1 Background

Before presenting and explaining the SLR, it is important to provide some explanation regarding the main topics of this paper.

2.1.1 (Business) Process Management

Business Process Management "is, in essence, a management idea. Organizations perform better when they pay explicit attention to their business processes from start to end ... To do this well, it is essential to understand the steps that are ... part of a business process, ... the people who are involved in these steps, the information that is being exchanged and processed ..., and the technologies that are invoked when executing the various steps." (Reijers, H.A., 2021) In other words, it uses a set of activities or tasks, web services, processes, layouts, and others, towards a specific goal, for example, the payment of a procedure.

2.1.2 Low-code Approaches

Low-code, as mentioned before, is a "visual approach to software development, " (mendix,) which means that "you can abstract and automate every step of the application lifecycle to streamline delivery of a variety of solutions" (mendix,). This means that programmers do not need to be highly specialized, which can be hard to find and costly, and the development of the application is usually faster, more interactive, and more agile. Due to the visual aspect, stakeholders can better understand what is being created and propose changes accordingly (instead of waiting for the "final stages of development" to propose changes) (mendix,). Low code can also be very useful for process automation for the same reasons mentioned earlier.

2.2 SLR 1: Current DT representations

This SLR aims to respond to the following questions:

1. What is the current state of art/frameworks/reference architectures used in digital transformation, preferably in government and related to process/workflow automation?
2. What are the main problems and difficulties regarding digital transformation and process/workflow automation in the government?
3. How are approaches and technologies (preference in low-code ones) used in government entities' digital transformation and process/workflow automation?

Since both SLRs follow the same steps, those will only be explained once, unless they differ from the other SLR:

1. Search Process, which is where documents from the Scopus database, that followed the search query " (Model OR framework OR (reference AND architecture) OR (state AND of AND art) OR (state-of-art) AND (((process OR workflow) AND (automation) OR (digital AND transformation) OR (applications AND ((low AND code) OR low-code))) AND (government OR public) were retrieved;
2. Inclusion-Exclusion Criteria, which is where the papers were screened and evaluated as relevant or not relevant;
3. Data Extraction, which retrieved the information relevant to answer the questions proposed and other information;
4. Quality Assessment, where the papers were evaluated based on how well they answered the questions.

In summary, from the SLR performed (information presented in Figure 20):

1. Organizations are afraid to perform its DT because of the problems previously mentioned, adding to inexperience and need for training, (Singh, J. P. and Kattimani, S. M. and Kumar, R. and Ramanathan, S. and Prasad, G. N. V., 2018) and (Krishnan, G. and Ravindran, V., 2017), perceived risks like security, (Liu, C. and jun tao, X. and Lu, B., 2016), costs (Liang, P. and Liu, Z., 2010), lack of flexibility (Eckermann, O. and Weidlich, M., 2011), and interoperability issues, with relation to legacy technologies and methods (Coursaris, C. K., 2012), for example.
2. Governments are even more apprehensive because of the risks associated with large quantities of data that need to be stored and maintained, for example, from transactions between citizens and the government (mendix,).
3. Most of the papers obtained do not mention low-code, or it is unknown if the applications used contain some low-code elements.

2.3 SLR 2: Modules present in the DT of government processes

This SLR functions similarly to the other but now focuses on the modules present, as mentioned. The

query was ((Enterprise AND Information AND Management) AND (System OR Solutions)) AND Government AND ((Digital AND Transformation) OR (Process OR Automation) OR Digitalization OR Low-Code OR (Low AND Code))

State of the Art

From the SLR performed, the information obtained (and presented in Figure 1) is used to create the following state-of-the-art:

Paper Name	Modules
Is e-government serving companies or vice-versa?	Portal (including translation), Documentation.
Deriving user requirements from business process models for automation: A case study	Eletronic Document Management System, Web services (such as Payment), Legal Entity System, Central Civil Rights System, Entities, Communication, Roles.
Ad-hoc business process management in enterprises as expert communities	Business Process Models, Roles, Services.
A research on the enterprises information management based on e-commerce	Information Support Layer, Management Support (link between Databases and Application Layer), Application Layer, Services (such as Payment and present Visitors info), Databases, Communication, Legality.
The changing face of government: The trends and a solution architecture for citizen services	Channel communication (web/mobile) to G2C and G2E, Portal, Content Manager, Transactional Service, BP Choreography,, Database, Web Services (SMS), Agency System, Social Media component. Entities.
Discovering business models for software process management: An approach for integrating time and resource perspectives from Legacy information systems	Databases, BPMN Portal to use that information (mentions the need to transform data which is unnecessary by using IMS – Information Management Systems).
Mobile supported and process enabled electronic document management system for local municipalities	Document Management, e-Government Portals, Notification Process, Client Layer, Server Layer (WebServer, Service Server, Database Server), legality, Roles, Entities.
The e-Government interoperability through Enterprise Architecture in Indian perspective	Portal (GEAF) uses Internet to Citizen, Online Transactions, Database, communication through NSDG.
Supporting legal requirements in the design of public processes	Communication, Legal Requirements Module, Roles, Entities.
Business process management: A holistic management approach	Business Process Models, Regulations, Communication, Documentation/Content, Roles, Permissions.
Modeling of information System for Licensing and Extension of Special Hajj Pilgrimage Organizer	Documentation, Network connection between Government Departments, Portal, Web Services (such as add documents and track status), BPMN.
Enterprise Content Management and the Records Continuum Model as strategies for long-term preservation of digital information	Documents, records, communication, requirements
A dynamic-capabilities view of local electronic government: Lessons from two successful cases	Web Services (such as Payment), Portal that contains E-services (Procedures/Transactions), Communication, Monitoring, Mobile.
Proposal for application of data science methods in E-Government: A Case-study about the application of available techniques for performance measurement with the help of data science	BPMN, Databases, Portal, Roles, (Web) Services, Documentation, Communication, Roles.
Towards Setting Up a Collaborative Environment to Support Collaborative Business Processes and Services with Social Interactions	User application, BPMS layer, Integration Layer, Social Media, (Web) Services (such as Notifications, Login, Creation and Viewing of process), Documentation, Web Site portal, BPMN.
Service Oriented Design for Indonesian E-Government System Using SOA	Processes, Core Services (such as payment), Portal, Database, Documents (Files), Communication, Entities, .
Intelligent information management with digitization workflow	Security methods (Face Recognition), Documents/Records Databases, Communication, SaaS user interface, Regulatory Compliance, PaaS Process service and back office service, Portal for Content Distribution (such as Internet or E-mail).
Codezin: braving the startup storm	Web Portals, BPM Models, Communication (with users through E-mail, social media), (Web) Services (such as Notification).
MARREG - MARriage REGistration [for a better cause]	Portal, Web Services (such as Payment and Notifications), Communication (as E-mails), Document gestion, Databases, Compliances.

Figure 1: Modules present in each paper from the second SLR

1. Each web page should have different security/permissions, (Aysolmaz, B. and Demirörs, O., 2014), to restrict each employee to their specific task/role. The majority of the papers contain the portal/user page, such as (Mazumder, A., 2020) that mentions an " eGovernance application software that is accessible through a portal by citizens" or (Paul, A. and Paul, V., 2012) that explains that the stakeholders interact with the National portal through the internet and that portal connects to the service providers. Another common trait is that most of them mention the citizen's interaction/web pages and not the employee's, which this paper focuses on.
2. Some entities contain a lifecycle, and documentation, even if it can also be stored independently. (Aysolmaz, B. and Demirörs, O., 2014) mentions the creation of an "automatically generate user requirements document". This automation is important and should be incorporated into the lifecycle of the process entity.
3. There are also other components, such as web components, that should be stored in a different directory. Examples of communication technologies are web services, which are vital in business process management, and low-coding approaches, such as when sending emails or completing payment, as cited by (Mazumder, A., 2020) that mentions a payment service done "through e-payment receipts portal".

A brief comparison between EIM systems is also performed in Figure 2.

	BPM (Business Process Models)	Entities	Collaborative Workspace	User Interface/ Homepages	Lifecycles	Security	Roles	Web Services	Organizations	Mobile Support
OpenText AppWorks	x	x	x	x	x	x	x	x	x	x
OutSystems	x	x	x	x	x	x	x	x	x	x
Oracle BPM	x	x	x	x	x	x	x	x	Unknown	x
IBM BPM	x	x	x	x	x	x	x	x	Unknown	x
SAP BPM	x	x	Unknown if collaborative	x	x	x	x	x	x	x

Figure 2: Table Showing what standard low-code components each technology solution contains

From it, is understood that most low-code solutions use the same components, is up to each government (or company responsible) to choose what better suits them.

3 Use Cases

Each government is different, having different ways to resolve and deal with its processes. Because of this, and since there are multiple use cases possible, this chapter only presents the ones deemed most relevant, given the SLR 2 and the original real-world implementation.

Figure 3: Use Case Diagram 1: Resolve a Procedure

The use case Resolve a Procedure, Figure 3, is the main one, appearing the most in the papers identified from the SLR performed, such as (Mazumder, A., 2020), which mentions it when performing and

resolving the marriage certificate. The goal is to resolve a procedure, even if they have different types. To achieve it, first, the employee (Revision Group) needs to fill in the necessary information and send it to another employee (Resolution Group). This employee will then decide if the procedure's resolution is positive or negative, which is then finalized through several automatic activities (such as generating documentation relevant).

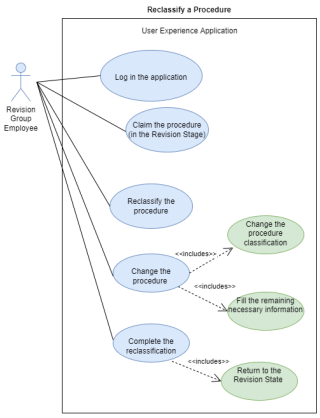


Figure 4: Use Case Diagram 2: Reclassify a Procedure

The other use case is Reclassify a Procedure, Figure 4 which is when a procedure is not created or classified correctly. In this case, the Revision Group Employee sends the procedure to the Reclassify state, where it will be changed and sent back to the Revision state.

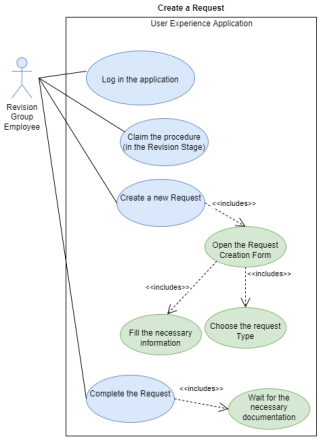


Figure 5: Use Case Diagram 3: Create a Request

Create a Request, Figure 5, is the use case where more information is necessary, for example from the citizen or an employee. In that case, the Revision Group Employee creates a new request, opening the Creation Request Form and choosing the type of request (internal, external, or citizen), filling in the necessary information. Upon receiving it (such as documentation), the request is completed by the employee.

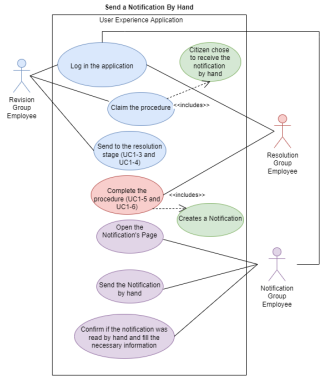


Figure 6: Use Case Diagram 4: Send a Notification By Hand

The last use case identified is the Send a Notification By Hand, Figure 6, which is when the notification needs to be sent through mail or a carrier instead of digitally. After the procedure's resolution is completed, a notification is created automatically. If the citizen chooses to receive that notification by hand, the Notification Group Employee will claim it and proceed to its sending, by filling in the necessary information. Upon receiving the confirmation, the Notification Group Employee fills in the remaining information and completes the notification.

4 SOLUTION PROPOSITION

This chapter presents the solution proposed in this paper, which is divided into 3 parts: Domain Model and Lifecycles, Reference Architecture, and Application Implementation.

4.1 Domain Model and Lifecycles

This section is divided into two, the domain model, explaining the main entities and their relationships, and the entities' lifecycles, detailing the different states and transitions.

4.1.1 Domain Model

The Domain model, Figure 3, contains four main projects which are Common Parts, Procedures, Notifications, and External Entities.

They are named projects because it is the name given by the low-code application used, AppWorks, to designate the different parts (directories) of the application.

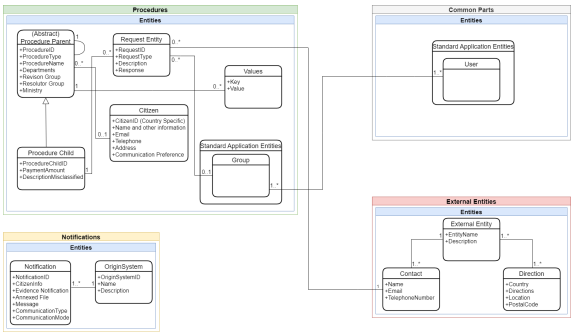


Figure 7: Domain Model

The Procedures Project contains the entity Procedure Parent. This is the main entity, which is abstract, and refers to the government procedures that will be dealt with by the employees. It contains the main properties such as procedure ID, type, the different departments, groups, and others.

Important to refer that not all properties in this and other entities will be presented because they are dependent on the specific government's applications and cases, or because they are system properties (internal).

The Procedure Child serves as the realization of the Procedure Parent, inheriting it, and adds a few new properties such as procedure child ID, procedure type child, descriptions, and payment amount.

To simplify future mentions of Procedure Child and Parent, they will be referred to as a whole, named Procedure.

There is also the External Entities project, which contains the information relative to each Request from the Procedures entity. External Entities contain the entities External Entity, Contact, and Direction.

Both the Common Parts project and Procedures contain Standard Application Entities, which are entities not created by the developers but by the application, internally. An example of them is the entity Group, which allows the employees to be grouped, such as Revision or Resolution as mentioned before, allowing them to have specific permissions and authorizations.

In the Notifications project, there are two entities, Notification, and OriginSystem. OriginSystem represents the system from which the notification was initially created (employee's application, citizen's, or other). There is also the Notification entity, the main entity here, that refers to the notifications created.

4.1.2 Entities' Lifecycles

Some entities can contain a lifecycle, which comprises their different states and transactions, especially if they have tasks that need to be completed, either manually or automatically. Regarding the ones mentioned, only Procedures (Procedure Child), Request Entity, and Notification contain them, which will now be mentioned.

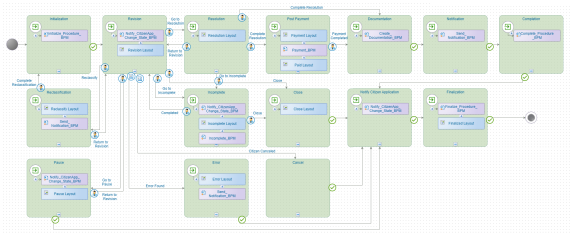


Figure 8: Procedure's Lifecycle Model

The Procedure starts in the Initialization state, which is where the process to initialize a procedure is done. It comprehends from retrieving the procedure ID from the citizen's application, the confirmation that the procedure was paid, attributing the read/write permissions, relative to the type of procedure, and others.

From this state the procedure goes to the Revision, where the Revision Group employee verifies the information referent to the procedure, adding new information if necessary. From this state, the procedure can go to various paths, including the main one, Resolution, and the alternatives, Reclassification (wrongly classified), Error (error found), Incomplete (missing information or documentation from the citizen or other employee), Cancel (procedure canceled by the citizen), Close (procedure closed by the employee), and Pause (which can be achieved either automatically or manually).

In the Resolution State, the employee from the Resolution Group decides if the procedure has a positive or negative resolution.

From this state, the procedure can go to the Post Payment one (if further payment is required), the Documentation state (where the documentation related to the procedure is created), or the Incomplete state (Resolution Group employee identifies missing information, sending it back).

In the Notification state, a notification is created and sent to the citizen. In the Completion State, the procedure is completed (in the Citizens' application), notifying that the procedure was completed correctly or incorrectly, and performing the last property updates such as date completion.

The last two states, common to almost all states are the Notify Citizen Application state, which is where the citizen is notified of the (previous) state (for example, if an error occurred which cannot be resolved, the citizen is notified of that) and Finalization where the procedure in finalized in the Employees'

application, removing the ability to modify the procedure and its documentation (converting all its permissions to read-only).

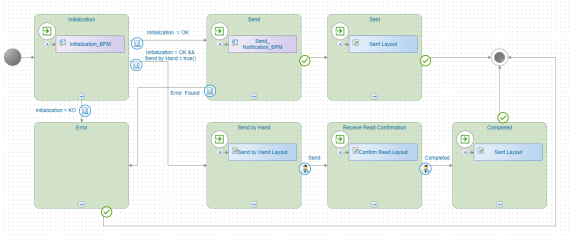


Figure 9: Notification's Lifecycle Model

The Notifications entity starts similarly to the Procedures (Initialization) but branches into 2: or it can go to the Send state, where the notification is prepared to be sent to the citizen or the Send by Hand if the communication preference chosen is to receive notifications by hand.

From the Send state, it goes to the Sent, where the notification is confirmed correctly sent and its lifecycle finished.

In the Send by Hand, the Notification Group employee revises and confirms the sending of the notification (use case 4). From there, it goes to Receive Read Confirmation, where upon receiving the confirmation that the notification was read, the employee inputs the final values such as read date and confirms its completion. The following and last state is the Completed state, where the notification is confirmed read, and completed, finishing the lifecycle.

There is also another state, achieved from the Initialization, and Send, which is the Error state, signaling that some error occurred, and aborting the sending of the notification.

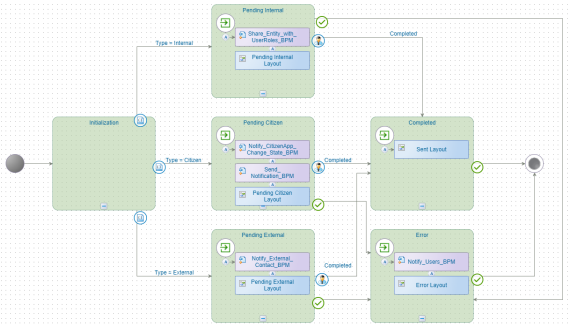


Figure 10: Request's Lifecycle Model

The Request's lifecycle also starts in the Initialization state, where the identification of its type is done. If Internal, it goes to the Pending Internal state, which communicates with the employee or chosen group. If External, it goes to the Pending External, where is done the communication with the external entity. Finally, if Citizen, it goes to the Pending Citizen, which will communicate with the citizen related to the procedure.

If everything went correctly, the request, from all the Pending states, goes to the Completed state, which represents the request sent, finishing the lifecycle. If not, the request goes to the Error state, finishing its lifecycle and not performing the request.

4.2 Reference Architecture

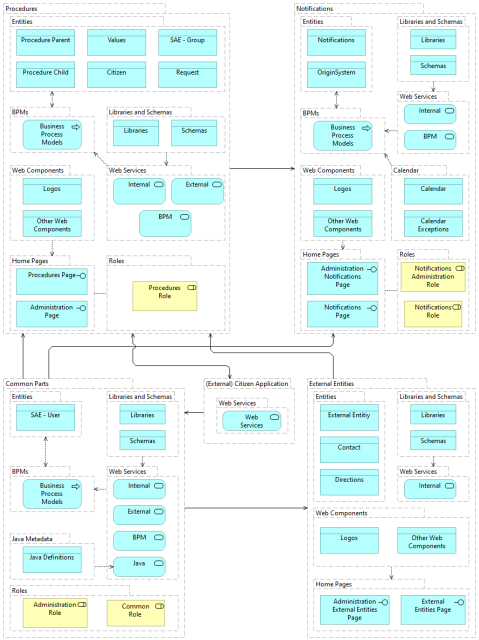


Figure 11: Reference Architecture

This reference architecture contains 5 projects:

- Common Parts
- Procedures
- Notifications
- External Entities
- (External) Citizen Application

4.2.1 Common Parts

The first project, Common Parts, was created with the goal of having all the components that are common to the various projects in a single location, in a way that eases their usage and avoids circular dependencies, (Russo, M. and Ferrari, A.,). This project contains the module/directory Entities, presented in the previous chapter. It also contains the Business Process Models (BPMs), which are the models used to implement the government's business processes.

The BPMs use the Web Services module, which contains 4 types of web services:

- Internal: if created by the application, such as predefined in an entity (read Entity, update Entity...)
- External: Retrieved from other applications (Citizen...)
- Java: Web services created from the Java Metadata module
- BPM: Might be specific to the program used. When creating a Business Process Model it is possible to generate a web service from it, making it possible to be used in other projects or applications/programs

Web Services use the Libraries ("collections of prewritten code that users can use to optimize tasks", (Ryan, 2020)) and Schemas modules ("a formal description of the structure or organization of a particular database", (Dearmer, A., 2021)). These can be internal if predefined by the application or external.

Another module present is the Java Metadata, which contains the java definitions. Every program/service that is created externally in Java (or another language if that is the case) is stored here. Normally, these web services are created only if they do not exist in the application used.

Finally, there is the Roles module, containing the roles created, and grouping the employees. In this project, Common Parts, there are two, the Administration Role which receives more permissions, and Common Role which contains more generalized ones.

The following projects will not be explained in so much detail, only being presented the differences and important points.

4.2.2 Procedures

Regarding the Procedures project, they have the same modules as the Common Parts but now the Web Services module does not contain the java BPMs (since they are in the Common Parts). This project also presents a new role, Procedures Role, which gives permissions related to procedures (read, write...).

This project also contains two modules that were not present in the previous one, which are the Web Components and Home Pages.

Home pages are the front end of the application, where the employees can create and resolve procedures. It contains two home pages, Administration Page, reserved only for administrative tasks and the Administrative Role, and the Procedures Page, accessed by the Procedures role users. These home pages can use web components which are, for example, images related to the government.

4.2.3 Other Projects

The Notifications project contains a new role, the Notifications role which is responsible for the notifications, and two new home pages, the Administration Notifications Page and Notifications Page (which is where the tasks to complete the notification's lifecycle are done). Notifications also do not contain external web services because they are already created in the Common Parts.

External Entities project contains a new home page, Administration and External Entities pages, and does not contain Roles or BPMs (especially because it does not have an entity with a lifecycle). The only web services present is internal.

Lastly, there is the (External) Citizen Application, which is a project not created in the employees' application but refereed in this reference architecture to represent the citizens' application. Here, only the web services are relevant, being used in the other projects.

4.3 Application Implementation

Before explaining how to implement the information presented using low-code technologies (in this case AppWorks), first, it is necessary to mention two other EIM solutions, which are Active Directory (regarding OpenText is called Directory Services, OTDS, (OpenText,)) and Content Management (in this case Content Server, OTCS, (OpenText,)).

OTDS allows users, in this case, the employees, to have a single sign-on on all the OpenText Applications. More importantly, it contains the possibility to group the users and transfer that information to AppWorks, relating it to the roles module.

OTCS refers to content management. Every document created is stored in this application, which is connected to AppWorks.

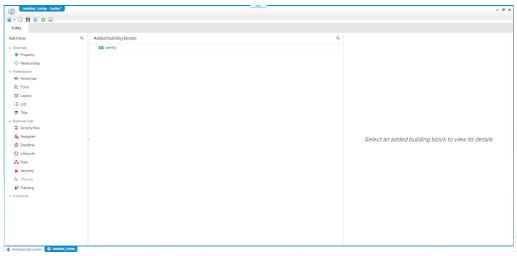
Since the majority of the EIM solutions, such as OutSystems (OutSystems,), contain both these applications, what is mentioned here can be applied to them.

Regarding AppWorks, to create the various projects (Procedures, Notifications, and so forth), first, it is necessary to configure and create an organization. Organizations or tenants are containers for "items of your organization such as users, domains,..." (Saxton, A.,). Inside an organization, it is possible to create a Collaborative Workspace, which is where "all modeling activities are done" (OpenText,). This is where the developers can work simultaneously, functioning similarly to a GitHub repository.

After creating each project, it is necessary to create the different modules that they contain (such as

entities or home pages). This is done through a file system where each sub-folder, from the project, corresponds to each module mentioned.

The folder Entities contains all the entities present in the Project, with each one containing various building blocks, Figure 12.



1

Figure 12: Entity Object

Explaining a few of them:

- Properties: Properties belonging to an entity such as Payment Amount or Departments.
- Security: Each entity can have certain permissions to create, view, and utilize them, which depend on the user's roles.
- Lifecycle: "Adds case management functionality to an entity" (OpenText,). Case management helps "manage...unstructured and typically most expensive and complex processes" (OpenText,). Entities' states and transitions, mentioned earlier.
- Web services: Include the ones related to the entities, BPMs, or other components/modules.
- Business Workspace: Enables the connection to the Content Server, allowing the entity to have documentation associated (OpenText,).
- Email: "Adds the ability to send and receive emails associated with an entity instance". It is what allows communication with citizens, external entities, and other employees...

The transformation from the Domain Model to the AppWorks is direct, only requiring the creation of the Entity and the addition of the necessary building blocks. Specifically, the building block Lifecycle, Figure 13, can use the information regarding lifecycles to ease its creation, given that the modulation used to present them is the same used in AppWorks.

Regarding the BPMs module, the information provided does not go in-depth because Business Process Management is used to model "well-defined, structured, processes" (OpenText,). Meaning that they are private and specific to the Government.

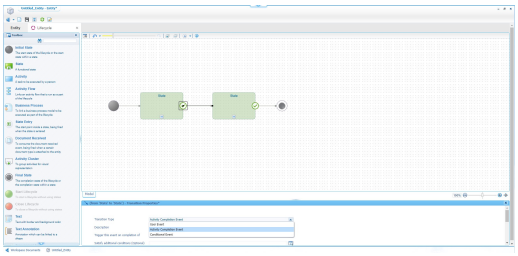


Figure 13: Lifecycle Editor

From the Web Services module, the four types and the information on how to create them (except for the Internal ones which are created through the Entity) were already provided.

Another modules present are the Home Pages and Web Components. AppWorks is divided into two parts: the AppWorks Platform, which is where the CWS, projects and other components are. This is the developing/administration part (the back end). There is also another part called AppWorks Experience (the front-end), Figure 14, that corresponds to the user interface that employees use to resolve the procedures.

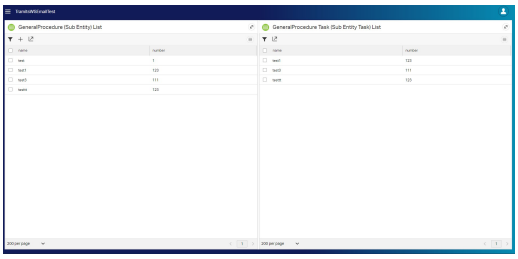


Figure 14: AppWorks Experience

Home pages can also use external components, from the Web Components module, such as Web Content (link to URL pages like Google Maps) and also pictures, namely Government logos or watermarks.

The last module mentioned is Roles. Each user can be assigned one or more roles in the application through a specific component called "User Manager". This component contains the roles standard in the AppWorks (administrator, user...) but can also contain roles created inside the projects. These can have certain permissions, which can restrict the actions that the user performs (such as access to the entity, or web services).

With this explanation regarding how to proceed with the implementation, the idea is also to show that low-code technologies (such as AppWorks) allow the creation of applications faster and easier than using "traditional software", which can help mitigate the various concerns that governments have regarding DT.

5 Evaluation

Since the development of the software was done in a government context, the evaluation must be qualitative and not quantitative. This is because the government wishes to remain anonymous, which also comprises the testing of the application, from the employees. Given this and other limitations such as the topics mentioned being niche, there are not many participants that can evaluate what is proposed.

Having that clarified, this evaluation is divided into 3 parts:

- The first part consists in interviewing and questioning Juniors and Experts regarding the models and reference architecture here presented.
- The second part questions the people involved in the real-world application that this paper is based on. They are also divided into Juniors and Experts.
- The last part relates this paper with other two similar, retrieved from the Related Work chapter, questioning, for example, if this paper could have helped them.

In the first two parts, the majority of the questions were converted to a five-point scale, with 1 being the lowest score, and 5 the highest, (such as if the information provided was useful enough) to provide a statistical evaluation despite the number of participants being very low.

5.1 Experts/Juniors Evaluation

This evaluation was performed by doing an interview with five participants (three experts and two juniors), where they answered five open questions:

1. Is the information presented here enough to give a ”head start” or to aid in the DT of governments?
2. How easy it is to interpret what is mentioned here, from a junior’s perspective?
3. Are the models and reference architecture flexible and scalable enough to be adapted in real-world scenarios?
4. What are the main limitations and flaws in the models and reference architecture presented?
5. Would you use what is here presented in your next DT of processes?

Figure 15 presents the findings regarding the questions asked:

In summary, experts mention that the models and reference architecture presented allow scalability and flexibility enough to mold specific use cases. Although the information presented is not enough, it is useful to prevent various problems, help juniors create project demos for governments, and to better explain the government personnel what are the goals of the project.

Questions	Juniors	Seniors
1. Is the information presented here enough to give a "head start" or to aid in the DT of governments?	• Mention that despite lack of knowledge, the information presented is like the real-world scenarios regarding the DT of government processes.	• Although the information presented here is not enough to create a full fleshed project, it is useful as a base, and it helps juniors create project demos for governments; • It is also useful to better explain governments what the DT consists.
2. How easy it is to interpret what is mentioned here, from a junior's perspective?	• Despite not having much experience in this type of transformation, the information is easy to understand and to follow (this can be extrapolated to government personal).	• Not asked.
3. Are the models and reference architecture flexible and scalable enough to be adapted in real-world scenarios?	• Not asked.	• The information presented allows scalability and flexibility enough to mold their specific use-cases and adapt the RA and models provided; • Few DTs cannot fully use the models and reference architecture provided.
4. What are the main limitation and flaws in the models and reference architecture presented?	• Mention the lack of specification, mostly and detail, for example in the BPMs or in the lifecycles regarding the transitions.	• The information provided is not "new", but it is good to see it modeled and researched since they are not aware of similar studies; • Mention that not every DT can take fully advantage of low-code technologies.
5. Would you use what is here presented in your next DT of processes?	• They would use it, because of all the benefits already mentioned.	• They would recommend it to Juniors, for example when performing demo projects to show to the clients; • They would use it to make the DT more standardized (with the adding bonus of preventing problems that sometimes occur).

Figure 15: Experts/Juniors: Summary of the Findings

Regarding the questions’ scores, present in Figure 16, the mean obtained was 4.04 with a Coefficient of Variation = 6.79454%, approximately 4 out of 5, meaning that the models and reference architecture presented to the participants are indeed relevant and useful to them.

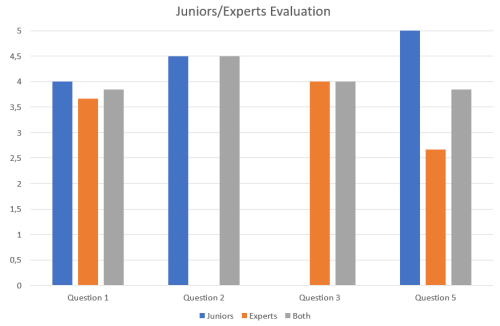


Figure 16: Scores from the answers to the questions proposed

5.2 Developers Evaluation

This evaluation focuses on the developers (four participants, two experts, and two juniors) responsible for the creation of the application, that similarly to the participants from the previous sub-section, answered five open questions in an interview, which were:

1. How long did it take and how easy was it to create the application from start to finish?
2. What were the limitations or problems faced when using that software, and how easy was it to solve them?
3. Regarding the feedback-oriented approach, what were the main advantages provided by using the

software low-code?

4. If given a choice, would you choose to use the low-code software or resort to a more ”traditional” way?
5. After reading the models and reference architecture here presented, do you think it would have aided in the creation of the application? How?

Figure 17 presents the findings regarding the questions asked:

Questions	Juniors	Seniors
1. How long did it take and how easy was it to create the application from start to finish?	• Development took roughly a year (this first phase, which the company was responsible for)	• Development time is deemed fast (compared to using "traditional ways"); • They note that the project changed a lot over the time, which was made possible and facilitated because of the usage of low-code technology; • Refer also that the very few human resources were necessary, only requiring a group of 4/5 people.
2. What were the limitations or problems faced when using that software, and how easy was it to solve them?	• Report that the major problems faced were regarding the application used, such as bugs or limitations); • Lack of knowledge regarding the usage of the software, in a project of this magnitude	• Adding to what was said, not accounting for possible future problems, which originated circular dependencies, difficult to resolve; • Debugging, sometimes is hard because the details of the error are not specific.
3. Regarding the feedback-oriented approach, what were the main advantages provided by using the software low-code?	• The clients were satisfied because they felt inclusive in the development of the application, since they could provide a lot of feedback which was treated rapidly.	• The feedback oriented approach, although more demanding on the developers was better than just receiving it in the end, when making major changes could be complex, time consuming and so on.
4. If given a choice, would you choose to use the low-code software or resort to a more "traditional" way?	• They agree that using low-code software has a lot of advantages, and would use it again, given the choice; • Mentioned that, however, would still like to use "traditional ways" when necessary, if possible. • AppWorks is easy to learn, compared to the traditional ways, even if they allow a better understanding of the application (and better debug sometimes)	• Same opinion as the juniors, using low-code is preferred even more because the ability to integrate the traditional ways (programing languages) is possible, at least in AppWorks; • Management side, such as controlling users or processes is also easier.
5. After reading the models and reference architecture here presented, do you think it would have aided in the creation of the application? How?	• Both juniors and experts agree that having used the information provided would have accelerated the development of the project, since it would have mitigated the problems faced.	• Apart from what was mentioned, it would have allowed the juniors to faster and easier create a demo in the initial stages, allowing even earlier feedback.

Figure 17: Developers: Summary of the Findings

Summarizing, developers report that the development of the application took roughly one year, with the project changing a lot over, from the initial planning. The major problems were related to the application and its limitations (such as bugs or other issues) and lack of knowledge regarding the use of the software. Since the development was faster and could be done in real-time with an interface, the clients were satisfied because they felt included in the development and could provide direct feedback, which lead to more changes in the application's goals and features. Both experts and juniors agreed that using low-code software to develop this type of application was very useful and they would choose to use it again if given the choice (even with the problems previously mentioned). It was faster, easier and despite the learning curve, allowed juniors to work semi-autonomous and without major flaws/errors. They note, however, that ”traditional ways” allow a better understanding of the processes behind the development and, sometimes, a better identification of the errors that appeared. A balance between using low-code technologies and ”traditional ways” would be ideal.

Regarding the questions' scores, present in Figure 16, the mean obtained was 3,94 with a Coefficient of Variation = 10.40863%, approximately 4 out of 5, meaning that, similarly to the juniors' and experts' evaluation, the usage of low-code technology and the feedback-oriented approach is very important and useful to the implementation of this type of DT.

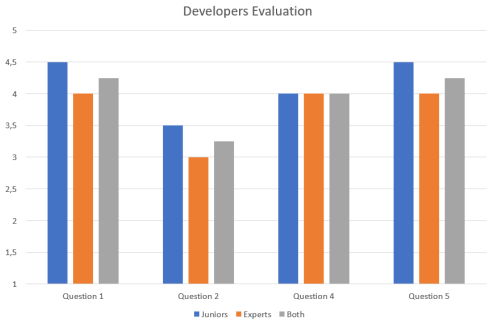


Figure 18: Scores from the answers to the questions proposed

5.3 Related Work Assessment

In this final sub-section, the goal is to compare this paper to others (from the related work) that mention similar DT, and, as mentioned, understand if they could take advantage of the solution proposed here.

The only two papers chosen were Mazumder A. (Mazumder, A., 2020) and Liu C. et al. (Liu, C. and jun tao, X. and Lu, B., 2016) because they are the most similar to this paper, containing a DT of processes and development of a real-world application. The idea is to:

1. Compare the theory and development to the other papers and extract information from them to enrich this paper or improve it in the Future Work chapter.
2. Understand if those papers could have used this paper to help them, for example, while developing the application.
3. Understand if their limitations could be fixed or mitigated with the usage of low-code software (if that is the case).

The Figure 19 presents the findings:

Questions	Paper 1: Mazumder A.	Paper 2: Liu C. et al.
1. To compare the theory and development to the other papers and extract information from them in order to enrich this thesis or improve it in the future work.	• Mentions practices and standards used, as well as the identification of the benefits of using them.	• Mentions the B/S architecture and the 3 tiers which could better group what was presented in this dissertation.
2. Understand if those papers could have used this paper to help them, for example, while developing the application.	• Helps them create the application and implement the theory.	• Helps them create the application and implement the theory; • Would have allowed a better explanation of the information presented with the AR and models.
3. Understand if their limitations could be fixed or mitigated with the usage of low-code software (if that is the case).	• The technology used to create the application was not mentioned but using the one presented here could provide all the benefits already mentioned.	• Many of the issues reported, such as lack of time or complexity could have been prevented by using the solution proposed here.

Figure 19: Related Papers: Summary of the Findings

The goal is to understand if this paper is useful to other scholarly papers and real-world scenarios, and if not, identify those flaws and correct them in future work. There are a few important parts that can be retrieved from the papers, such as from (Liu, C. and jun tao, X. and Lu, B., 2016) which mentions a B/S architecture that comprises the Presentation Tier, the Business Tier, and the Data Tier or (Mazumder, A.,

2020) that mentions that their application follows specific practices and standards which could be useful in future implementations of the solution proposed here.

Regarding this paper's contributions to those papers, they would have allowed not only a faster and simpler way to create the applications but also improved the feedback approach in its development. Several concerns such as data encryption or network security or even further functions like decision-making support and mobile office automation, could all be dealt with if the authors had followed the models and reference architecture presented here, and used a low-code approach when creating the application.

6 Conclusion

This chapter presents the main contributions of this paper and the future work that needs to be done in order to improve it.

6.1 Main Contributions

In conclusion, from the results previously obtained, this paper provides valuable information regarding the topic of digital transformation of government processes.

From the creation of a state of the art from an SLR regarding the main modules present in the DT of government processes to the identification of the most relevant use cases (Resolve a Procedure, Reclassify a Procedure, Create a Request and Notify Citizen By Hand), to the creation of the entity models related to the most important entities and their lifecycles, to the combination of that information in a reference architecture (adding other modules) and its realization through a low-code application.

All of this was done to provide information, aid in the DT of processes, and incentivize the governments to implement or improve their DT. However, since this project was tested in a government and the topics mentioned are niche in the field of IT, a full statistical quantitative evaluation was not possible. Nevertheless, it was realized as a qualitative one, which comprised a few interviews with experts and juniors in the field. Juniors are especially useful because their answers allow their extrapolation to the government personnel, which also lack technical knowledge. From the results, the major takeaways were:

1. The models and reference architecture presented are easy to understand and follow from an expert perspective but also a junior/low knowledge one.
2. The information here, despite not presenting new knowledge to the experts in the area, allows them to confirm their development, systematize the DT, and mitigate errors. It is also useful to explain this type of DT to juniors or people with not much knowledge regarding these topics.
3. The usage of low-code technologies eases the DT of government processes and allows the mitigation of issues that can come from its implementation.

Given these conclusions, it is safe to assume that what is here proposed helps solve the main issues when performing the DT of governments, which are the lack of a reference architecture and associated models and the lack of information regarding the usage of low-code technologies when implementing such transformation.

6.2 Limitations and Future Work

Regarding the limitations, several were already mentioned. One of them is that these models and reference architecture, do not go in-depth (such as providing more information regarding the possible BPMs or Libraries/Schemas). This was done on purpose since the idea was to provide a generalized view that could be used and molded in various cases. However, the lack of that detailed information makes what is presented here not as relevant to experts in the area, given their experience with this type of DT.

Another issue is that the information provided and its implementation are optimized to be used by low-code technologies, specifically OpenText AppWorks. This is because of the nomenclature and the different models, that although can be adapted to other technologies, were created with that specific technology in mind. The solution proposed is also very specific to this type of DT and might not be useful if that DT diverges a lot from this one. It is important to mention that further testing is necessary to improve the information presented here.

Regarding future work, an improved evaluation should be performed. As mentioned before, a better evaluation was not possible because it was being implemented in a government (that wishes to remain anonymous) and because it was specific to a niche field, making participants scarce, which in turn made the evaluation qualitative and not qualitative.

REFERENCES

Aysolmaz, B. and Demirörs, O. (2014). Deriving user requirements from business process models for au-

tomation: A case study. In *2014 IEEE 1st International Workshop on the Interrelations between Requirements Engineering and Business Process Management (REBPM)*, pages 19–28.

Coursaris, C. K. (2012). Enabling mgovernment: a framework and a case study. *International Journal of Electronic Finance*, 6:79–101.

Dearmer, A. (2021). Complete guide to database schema design. Available: https://www.integrate.io/blog/complete-guide-to-database-schema-design-guide/. [Accessed September 10, 2022].

Eckermann, O. and Weidlich, M. (2011). Flexible artifact-driven automation of product design processes. *Enterprise, Business-Process and Information Systems Modeling*, page 103–117.

Kimberling, E. (2021). How amazon is a roadmap for digital transformation success. Available: https://www.thirdstage-consulting.com/amazon-roadmap-for-digital-transformation-success/. [Accessed July 13, 2022].

Krishnan, G. and Ravindran, V. (2017). It service management automation and its impact to it industry. In *2017 International Conference on Computational Intelligence in Data Science(ICCIDS)*, pages 1–4.

Liang, P. and Liu, Z. (2010). Realization of administrative organs automation system ased on b / s mode. In *2010 International Conference on Machine Learning and Cybernetics*, volume 5, pages 2270–2275.

Liu, C. and jun tao, X. and Lu, B. (2016). Toward integrated and automated management of government affairs. *International Journal of Hybrid Information Technology*, 9:267–278.

Mazumder, A. (2020). Marreg – marriage registration [for a better cause]. In *2020 IEEE Integrated STEM Education Conference (ISEC)*, pages 1–8.

mendix. What is low-code? Available: https://www.mendix.com/low-code-guide/. [Accessed August 19, 2022].

OpenText. Opentext appworks platform. Available:https://www.opentext.com/products-and-solutions/products/digital-process-automation/appworks-platform. [Accessed July 15, 2022].

OpenText. Opentext appworksplatform architecture. Available: https://developer.opentext.com/. [Accessed July 21, 2022].

OpenText. Opentext case management. Available: https://www.opentext.com/products-and-solutions/products/digital-process-automation/alternative-process-solutions/opentext-case360. [Accessed July 21, 2022].

OpenText. Opentext content services platform. Available:https://www.opentext.com/products/content-services-platforms. [Accessed July 15, 2022].

OpenText. Opentext document management. Available:https://www.opentext.com/products-and-solutions/products/enterprise-content-management/content-management/opentext-document-management. [Accessed July 15, 2022].

OutSystems. The low-code development guide. Available: https://www.outsystems.com/guide/low-code/. [Accessed June 30, 2022].

Paul, A. and Paul, V. (2012). The e-government interoperability through enterprise architecture in indian perspective. pages 645–650.

Reijers, H.A. (2021). Business process management: The evolution of a discipline. *Computers in Industry*, 126:103404.

Russo, M. and Ferrari, A. Avoiding circular dependency errors in dax. Available: https://www.sqlbi.com/articles/avoiding-circular-dependency-errors-in-dax/. [Accessed July 21, 2022].

Ryan (2020). What are libraries in programming? Available: https://www.idtech.com/blog/what-are-libraries-in-coding. [Accessed August 25, 2022].

Saxton, A. What is a tenant? Available:https://powerbi.microsoft.com/pt-pt/blog/what-is-a-tenant/. [Accessed July 16, 2022].

Singh, J. P. and Kattimani, S. M. and Kumar, R. and Ramanathan, S. and Prasad, G. N. V. (2018). Need aspect approval system for procurement of components used for indian satellite program. In *2018 4th International Conference on Recent Advances in Information Technology (RAIT)*, pages 1–6.

APPENDIX

Title	Q1		Q2		Q3	
	Related		Related		Related	
E-Government For Public Values Creation: A Systematic Literature Review	Yes	• SLR, following PRISMA • Conceptual framework for public values enhancement through e-Government projects	Yes, but only DT.	Savoldelli et al.: crucial factors in e-government context could differ over the time, different cultures or countries cause problems in the adoption of DT.	No	No mention of Low-code, even process automation or even technologies for that.
Digital transformation of public services and administration: A case study of Bulgaria	Yes	• Presents Bulgaria's e-Government Model • Strategic Framework • Mobile application	Yes, but only DT.	• Existing regulatory framework did not stimulate e-government development and even hindered the use of e-artifacts – interoperability was hard. • Paperless approach: EDIMS • SEGA – State of e-government agency is mentioned.	Yes	There is mentions of technologies to adopt e-government (RegiX – Inter-Registry Exchange Environment and EDIMS) but unknown if they are low-code or EIM systems.
Towards integrated and automated management of government affairs	Yes	• System architecture: Three-tier B/S Architecture of ASP.NET • Integrated Automated Management System of Government Affairs: Approval Flow, Login, Document Management, System Management	No, at least directly	Mentions problems like lack of network security management module, like data backup, antivirus and so on, which can be interpreted as e-government problems.	Yes	• Usage of Microsoft SQL Server 2005 as relational database management system • Microsoft Visual Studio 2005 as the integrated development environment of the ASP.NET based web applications (ASP, PHP, JSP). • Utilization of flowcharts
Digital platformisation as public sector transformation strategy: A case of Ghana's paperless port	Yes	• Development of transformational affordance framework – TAF • Platformisation strategy	Yes, but only DT.	Mentions the problem that DT is well established in the private sector but lacks knowledge regarding public one, even more in less developed countries.	Yes	• Digital platform is layered. • Unknown if low-code.
An architecture for facilitating two-way G2C relationships in public service delivery	Yes	• G2C channel and architecture	Yes	• Problems with G2C communication platforms like being limited.	Yes	There is mention of Social media, G2C platforms and web or mobile applications, although no low code related. There is also mention of automation mechanisms but not in depth.
Digital Transformation? A Primer for Practitioners	No, at least directly	Although it does not mention government directly mentions other pillar which are also important not only to government but other organizations: CE – Customer Experience Transformation, BP – Business Process Transformation, BM – Business Model Transformation and OT – Organizational Transformation	Slightly	One problem presented is that DX is not only regarding automation of processes that are manual. There needs to be a transformation with focus on people, the costumers' experience.	Yes	Several open source DX technologies are mentioned Fat Free CRM, InfluxDB, D3.js, R, TensorFlow, Keras, Serverless, Apache Airflow, Activiti, PrestaShop and OpenCart, although very briefly.
Flexible artifact-driven automation of product design processes	Yes, only PA	• Automated support of business processes by information systems can be seen as state of the art for many domains • BPMN, EPCs, UML activity diagrams are state of the art for business process discovery and design	Yes	• The approaches regarding the previous question are not suited for highly creative processes, like in the field of innovative product design, requiring more flexibility.	Yes	• Use of WS-BPEL and WfMC reference model • ADEPT systems • The technology used, Oryx Project, Oryx Mashup-Framework is unknown if low-code approach or even easy to use.
IT service management automation and its impact to IT industry	Yes	• Mentions IT service management, a framework for enterprises to define, deliver, operate,..., IT services and meet business objectives. • IT service management as the implementation of IT services that meet business needs. ITIL is regarded as the best practices framework for implementing ITSM • Automation Scope Matrix	Yes	• Initial resistance is expected • Its critical to train human resources with advanced automation tools and techniques. • Requires enterprises to follow systematic approaches and processes for a successful implementation • Potential less jobs	Yes	• Mention of Basic automation, that with help from simples scripts and macros, automates repeated tasks by IT executives. Also mentioned other types of automation. • Mentions benefits of automation but unknown if low-code.
Enabling mGovernment: A framework and a case study	Yes,	• 3 stage approach: Process Transformation Framework, Mobile Content Platform, Asset Leverage Strategy – affords flexibility, scalability and mobile application • Mentions Content Management Systems • G2C, G2B (business), G2G, government's Internal Effectiveness and Efficiency (IEE) and overarching infrastructures (cross-cutting)	Yes	• One challenge is the, possible, lack of compatibility between mobile systems and existing eGovernment systems. Even harder if still using legacy systems. "The solution lies in implementing open systems using open, not proprietary standards"	Yes but not PA	• mGovernment – mobile government: use of mobile services • M-GOV project • Uses the proposed Process Transformation Framework but seems to lack features or be more specific.
The application of workflow technology in office automation system	Yes	• Mentions the Workflow Framework of COAS (Office Automation System but with Workflow), based on B/S structure.	No, directly	• There is no mention of process automation issues or challenges but office automation is mentioned several times. • Issues like large amounts of information make decisions regarding it more complex, costs associated, less flexibility...	Yes	There is mention of Microsoft Net Framework 3.0, Windows Workflow Foundation (WWF)
Need aspect approval system for procurement of components used for INDIAN satellite program	Yes	• Mentions G2E interaction, which relevant to my paper. • Mention of process flow diagram and architecture of the application used.	Yes	Mentioned earlier	Yes	Usage of 'e-NAC' approval system, a web based application software simulating workflow operations.
Deriving user requirements from business process models for automation: A case study	Yes	• Mention of framework ARIS • Entity Relationship diagram	Yes	Mentioned earlier	Yes	• Mention of PAIS, Process Aware Information Systems, developed as a workflow management (WMS) system or BPM system that executes operational processes based on formal models. • UPROM tool
Realization of administrative organs automation system based on B/S mode	Yes	• Presents a schematic topology diagram of civil air defense office automation system network • Workflow of internal examination and approval of incoming document process • B/S (Browser and Server) mode	Yes	• Complexity, heavy communication between government organs. Costs and limited trained staff are also important factors.	Yes	• Mention of Notes database and Notes server of Lotus, Domino • HTTP, HTML, URL, CGI, SSL protocols are integrated in the Domino server
Adopting DevOps in the real world: A theory, a model, and a case study	Yes	DevOps as a set of practices and cultural values that aim to reduce the barriers between development and operations teams.	Yes	Previously mentioned. Automation increases agility and reliability. It is necessary, however, to have test automation, quality checks.	Yes	Mention of ICT, information and communications technology.

Figure 20: Table containing information regarding the research questions from the first SLR