SWOT based Transformation'sOrganizational Risks'Management (STORM)

Antoine Trad, PhD
IBISTM, Paris, France

Damir Kalpić, PhD
FER, Zagreb, Croatia

INTRODUCTION

The Strengths (S), Weaknesses (W), Opportunities (O), and Threats (T) (SWOT) based Transformation Organizational Risks'Management (STORM) is crucial for the implementation of risky projects.Transformation projects depend mainly on thePolymathic-holistic Project Management Concept (PPMC); which means that SWOT is applied at all project's levels and components. From the strategy-definition's phase, going through the Requirements Engineering (RE) phase, and until the implementation phase, the PPMC must deliver the real-actual project status and risk evaluation outcomes. This is a complex task because it needsa cross-functional and cross-phase Decision Making System (DMS). The DMSfor PPMC should be designed to be used by the executive management, enterprise architects, business users, business architects, implementation developers, and other project actors. In this article, the authors propose the STORM and DMS based PPMC.The DMScan support managers in transforming the enterprise (simply*Entity)*, where they have to take into account, and address all possible risks, in order to enable the successof the complex *Project* Implementation Phase (PIP).Where thePIPconsiders the latest service technologies like MicroServices Architecture (MSA), Application Programming Interface (API), Service Oriented Architecture (SOA) and other relevant avant-garde technologies and topics. In general, managers have a siloed approach and impose separate PPMC'sphases, which may cause project's desynchronization and hence confusion. A PPMC usesRefinement Processes (RP) to support reverse/reengineering and integration of risk factors.In order to improve the project's success rates,the PPMC must adopt the optimal Enterprise Architecture's (EA)based transformation approach. The PPMCis not independent from any EA phase, where project teams must be capable to integrate PPMC and DMSinterfaces in their RP generated Building Blocks (BB) and Microartefacts. The PPMC ensures that the generated BBsand Microartefactsare independent of any specific methodology/technology, tool, brand, or other locked-in delimiter/actor. The project's in-house DMS is based on a heuristic evaluation model, which is presented in the Proof of Concept (PoC), and the applied business case. The PPMCcan be supported by the alignment of standards, methodologies, and development strategies, like the: The Open Group's Architecture Framework's (TOGAF), the Development and Operations (DevOps), and others… STORM offers, a set of recommendations which can applied by managers, enterprise/business architects, analysts, and engineers to implement solutions for transformation projects' strategy establishment. The PPMC uses an EA driven conceptthat uses sets of patterns-based BBs to support projects in selecting the right technologies and to enable an iterative change process.

Keywords: SWOT, PPMC, DMS, MSA, API Management, Transformation Initiatives, Enterprise Architecture, Development Cycles, Services, and Critical Success Factors.

PPMC'S APPLIANCE

This RDP's global topic is related to projects and in this continuous phase the Research Question (RQ) is: "Which STORM features, characteristics, and which type of PPMC should be used in the implementation phase of a transformation project?". There the Research and Development *Project* (RDP) is based on Risk management (like SWOT or others), RE, Architecture Development Method (ADM), DMS, Critical Success Factors (CSF), and Critical Success Areas (CSA). The Organizational and Digital Transformation Projects are very complex to finalize, and they depend on the RP. The RP of the legacy Business Unit (BU) which needs an In-House-Implemented (IHI) Methodology, Domain, and Technology Common Artefacts Standard (MDTCAS) that can map to any existing methodology or technology. The PPMC needs to define a MDTCAS manages RP's basic elements: BBs,Compound BBs (CBB), and Microartefacts. The major innovation in this article is linking of the Transformation Manager's(*Managers*)popular risk and quality management (like SWOT, Six Sigma, …) to CSAs, CSFs, and Key Performance Indicators (KPI); which in turn CSAs, CSFs and KPIs (simply *Factors*) link to concrete project's BBs.There the main topics are:

- Problem domain(s)are related to the transformation project (simply *Project*).
- Build a flexible and scalable Information and Communication System (ICS).
- RP based unbundling and restructuring strategy, delivers coherent sets of services, Microartefacts, BBs, and Solution BBs (SBB).
- Types and categories of BBs and SBBs to be (re)used.
- Possible real-world solutions and recommendations.
- The role of the *Manager* and team.
- PPMC and DMS, based on SWOT approach, which uses CSAs and CSFs.
- Supporting an *Entity* with PPMC.
- Linking STORM, hence SWOT to *Factors*.

This is a fairly complex and technical article; and a reader who wants to access just the main principles is advised to proceed to the PoC section.

Factors Management

SWOT risk analysis is a basic highly technical methodology that can be used to check*Project*'s strategy, capabilities or a BU'sviability,where the targetedAPplication Domain (APD) can be a local, or global activity. SWOT checks *Project*'s *strengths, weaknesses*, *opportunities and threats* or constraints, which can establish and *link-to* the initial sets of *Factors*; and that is a real challenge in the real world. Having the assumptions that:

SWOT Analysis = ∑ *Factors*, abstracts the risk on the level of a *Project*.

Factors = ∑ CSAs, abstracts the risk on the level of a subsystem or a sub-*Project*.

CSA = ∑ CSFs, abstracts the risk on the level of a PPMC component.

CSF = ∑ KPIs, abstracts the risk on the level of an BBs based SBB or a bundle of services.

KPI = ∑ Variables (VAR), abstracts, and attributes of a service(s).

The symbol ∑ relates to processing of a series of transformational equations, and not to the simplistic ***sumof***.. The linked sets of *Factors* can be external (***O*** and ***T***), or internal (***S*** and ***W***). SWOT is used for the *Project*'s preliminary activities, which is known as the ADM's preliminary phase [1, 2], but it can be reused in major *Project's* activities. Decisions for

formulating a *Project's* strategy are based on the analysis of the external and internal CSAs and hence CSFs and KPIs. *Strengths, weaknesses, opportunities, and threats* (or SWOT) is an *established* concept for the categorization of highly used CSFs. For example, an important CSF is how the project can achieve and sustain a significant APD or *business competitive advantage*; that can be done by using value chain analysis. The *Project's* strategy shows how this value can be developed and maintained [3]. To determine CSAs and CSFs, there is a need to review SWOT items which should reflect: 1) The *Project's* use of ***Ss***; 2) Eliminate possible ***Ws***; 3) Exploit ***Os*** (by using ***Ss***); and 4) Implement strategies to intercept ***Ts***. CSFs and KPIs are key elements in *Projects* and their planning. A CSA is a category (or set) of CSFs where in turn a CSF is a set of KPIs, where a KPI maps (or corresponds) to a single requirement and/or software artefact or a service (SOA, MSA, or other); a bundle of services is known as a Microartefact. For a given requirement, a feature, or a problem, the transformation team identifies the initial set of related SWOT elements, CSAs, CSFs and KPIs, for the use in the DMS.

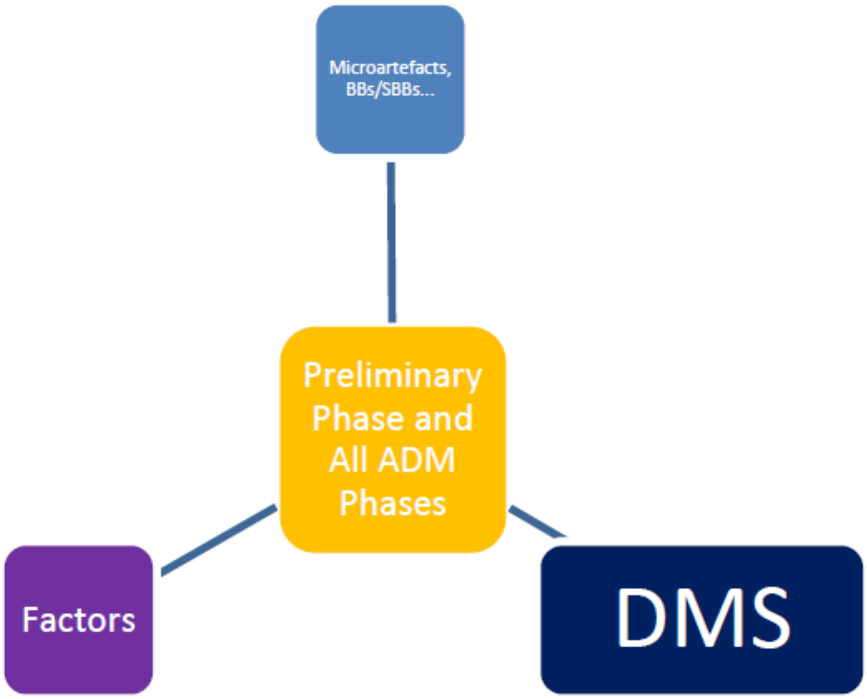


Figure 1. The relations between ADM's phases and other project's components

These *Factors*are mapped and they deliver sets of solutions or recommendations. Hence, *Factors* are important for the mapping between various types of *Project* artefacts, KMS knowledge constructs, Microartefacts, organisational items, and the DMS. Therefore, *Factors* reflect areas that must meet the main strategic *Project*PPMC and predefined (mainly financial) constraints. Gained knowledge/experience can be fed in the *Entity*'s DMS/KMS; and that is how it builds its own Iterative Learning Process (ILP) that is supported by the Applied Holistic Mathematical Model (AHMM) for PPMC (AHMM4PPMC).

The AHMM4PPMC based STORM/SWOT

The *Project* has to set the AHMM4PPMC based MDTCAS and ADM scopes and dimensions. Difficulties are due to the *Entity's* heterogenous parts. The AHMM4PPMC supports its feasibility and integrity.In this article the authors uses an adapted version of the AHMM4PPMC[54] to support STORM and PPMC feasibilities that uses the initial sets of *Factors*.The AHMM4PPMC supports iterative RP of the legacy systems, by using PPMC, MDTCAS and ADM to integrate standard methodologies, like TOGAF and ADM.For a *Project's* requirement orproblem, STORM identifies the initial sets of *Factors*, to be used by the Heuristics Decision Tree (HDT) based DMS4PPMC and maps these *Factors* to the sets of BBs/CBBs and requirements [55]. Hence *Factors* are important for the mapping between the requirements, ILP/knowledge constructs, RP generated artefactsand DMS4PPMC.So HDT's based evaluation processes/function (HDT.eval) can automatically estimate the values of *Factors*[56]; where STORM and SWOT is linked to concrete *Factors* to be used by the DMS4PPMC.

Linking STORM and SWOT to Factors

SWOT elements map/link to CSA by using the STORM2CSA structure. And Each CSA contains related CSFs and in turn KPIs where each KPI links to a concerted ICS variable (VAR); a VAR is a BB's attribute which can be presented as BB.VAR [49].

```
STORM2CSA
{
        S_Value         = HDT.eval( CSA.S_Value );
        W_Value         = HDT.eval( CSA.W_Value );
        O_Value         = HDT.eval( CSA.O_Value );
        T_Value         = HDT.eval( CSA.T_Value );
};
```

Linking a CSA to CSFs

CSA elements map/link to CSF by using the CSA2CSF structure:

```
CSA2CSF
{
        S_Value         = HDT.eval( CSF.S_Value );
        W_Value         = HDT.eval( CSF.W_Value );
        O_Value         = HDT.eval( CSF.O_Value );
        T_Value         = HDT.eval( CSF.T_Value );
};
```

Linking a CSF to KPIs

CSF elements map/link to KPI by using the CSF2KPI structure:

```
CSF2KPI
{
        S_Value         = HDT.eval( KPI.S_Value );
        W_Value         = HDT.eval( KPI.W_Value );
        O_Value         = HDT.eval( KPI.O_Value );
        T_Value         = HDT.eval( KPI.T_Value );
};
```

Linking a KPI to Business Scenarios

KPI elements map/link to VAR by using the KPI2VAR structure:

```
KPI2VAR
{
```

```
        S_Value      = HDT.eval( BB.VAR.S_Value );
        W_Value      = HDT.eval( BB.VAR.W_Value );
        O_Value      = HDT.eval( BB.VAR.O_Value );
        T_Value      = HDT.eval( BB.VAR.T_Value );
};
```

These Factors and ICS components are tuned through ADM's phases, as shown in Figure 1; and with that we establish an AHMM based transformation model[54].

The Transformation Model First Approach-Top Down Approach

ThePPMC is a*Model First Approach*that uses a pseudo bottom up (or a mixed approach) that is mainly based on: 1) EA based project-management;2) Managing BBs and Microartefacts, which result from the RP/unbundling-process; and 3) A middle mitigation process based on PPMC/DMS4PPMC. It is an agile upstream approach that accommodates to unbundle legacy services environments; and make them accessible using API's concept. Optimal PPMCs are derived from standardized EA methodologies, which in turn depend on the selected requirements' quality and their mapping to: *Factors* and BBs/Microartefacts. In PPMCwe do not refer hard links to services, but it will include abstract services that map to requirements, BBs/Microartefacts. This makes the *Entity* not locked in a specific APD/environment. In the PIP, the PPMC supports the services to requirements mapping process, by the means of abstract services. This ensures that projects are managed by the ADM and do use the pool of Microartefacts. The PPMC supports the alignment between Microartefacts/services, requirements, organizational (re)structure, AMD/governance phase(s), and the ICS. SBBs are used and interfaced by using a set of services and their integration's status can be queried by using Factors [4].

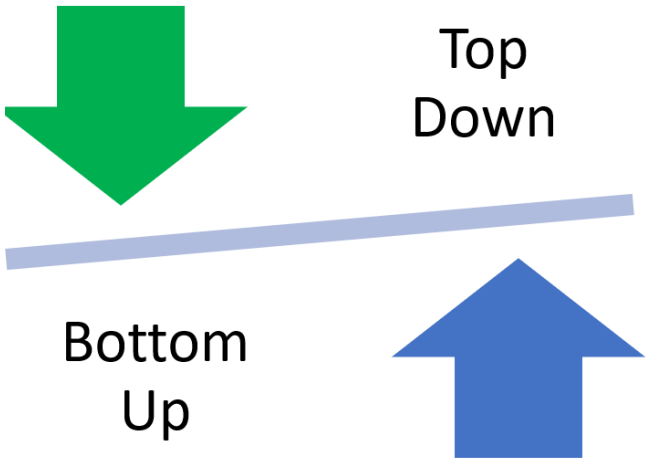


Figure 2. The transformation model-first approach.

To avoid problems in the complex PIP, the bottom-up approach is strongly recommended, where the 1st step is to convert the legacy system into a structured pool of BBs/Microartefacts and a repository of EA models. Parallel to that, as shown in Figure 2, this newly transformed structure needs an umbrella that is a high-level top-down EA management concept. It is observed that a "1:1" mapping approach would synchronize these two opposite approaches, and a basic unit of work is recommended to be defined. This unit of work links specific business (or non-functional) requirement in the following manner: 1) It describes the used PPMC and modelled Use Cases (UC) for a set of APD activities; 2) The PPMC makes the links to corresponding classes and diagrams; 3) To add the Microartefacts and EA models to

the project's architecture repository; 4) Refine, unbundle, and persist in directory for classification of the newly created Microartefacts; and 5) Use STORM to refine the strategy. The PPMC is a set of idioms and activities, where an idiom is a basic automation activity that is generic and not specific to any PIP; and reuses the unbundled BBs.

Unbundled BBs and RPs

The conversion of the*Entity's* legacy system(s), need an IHI PPMC and MDTCAS that map to existing Microartefacts, BBs, and CBB. In generating Microartefacts the RP can face major difficulties because of *Entity'*sheterogenous human profiles/cultures, system parts, managers/stakeholders exaggerated financial ambitions, and *Project's* limited time/budgets [5]. MDTCAS interfaces standard methodologies which are based on the Object Oriented (OO) Methodology (OOM) which have standard OO features, inherited fromRumbaugh, Booch, and Jacobson methodologies. The methodologies are the fundaments of the most known modelling/ICS standard, the Unified Modelling Language (UML) [6,52]. All methodologies like the ADM, are developed using an UML profile/metamodel. The first major paradigms that influenced MDTCAS are: 1) Rumbaugh's Object Modelling Technique (OMT), which develops manageable OO based SDCs and supports OO Integrated Development Environments (IDE). OMT's allows class attributes, methods, inheritance, and association to be coherently open to implementers; 2) Booch's methodology, focuses on OO Analysis (OOA) and OO Design (OOD) phases, and has five activities: Conceptualization, Analysis, Design, Evolution, and Maintenance of requirements and their related CBBs. It is cyclical (or spiral) model, which uses incremental implementation processes, which are the origin of the ADM and DevOps. OOA/OOD phases, use six types of models/diagrams: Class, State transition, Object, Process, Module, and Interaction, which all are MDTCAS basic artefacts. Class and module are static diagrams, while state transition are dynamic ones[6]; 3) Jacobson's methodology (OOSE) can be used to plan, design, and implement OO ICS components; and has five types of models: Requirements, used to specify Use Case (UC) diagrams, Analysis, Design, Implementation (used by RP), and Testing; they are PPMC MDTCAS's basic artefacts; 4) BBs, CBBs, and Organizational Process Models (OPM); and 5) UCs help the PPMC to analyze and extract BBs, CBBs, and the interaction between them to create OPMs. Where a UC can include: OOM diagrams, non-formal code, Events flow, Pseudo-code, and Actors. OOM, UC are the basis of the actual EA modelling languages to support BBs and CBBs to be used by the PPMC. BBs and CBBs can map to *Application Services*, where a CBB has the following types of resources: Business, and System or non-functional. CBBs can be modelled with *Business Services*, and a subsequent set of diagrams, BBs, *Application Services*, and others. When CBBs are refactored/identified as MDTCAS artefacts like composite application services, which can be used to build OBBs as shown in Figure 3. A *Project*needs a well synchronized ADM, in which the OPMS provides the support business, EA models, to enable the PPMC. That all needs a Polymathic-holistic approach to enable structured OPMs. Automated and non-automated OPMs have a key role in developing APD competencies, and where *Business Architecture* and ICS architecture are vital.

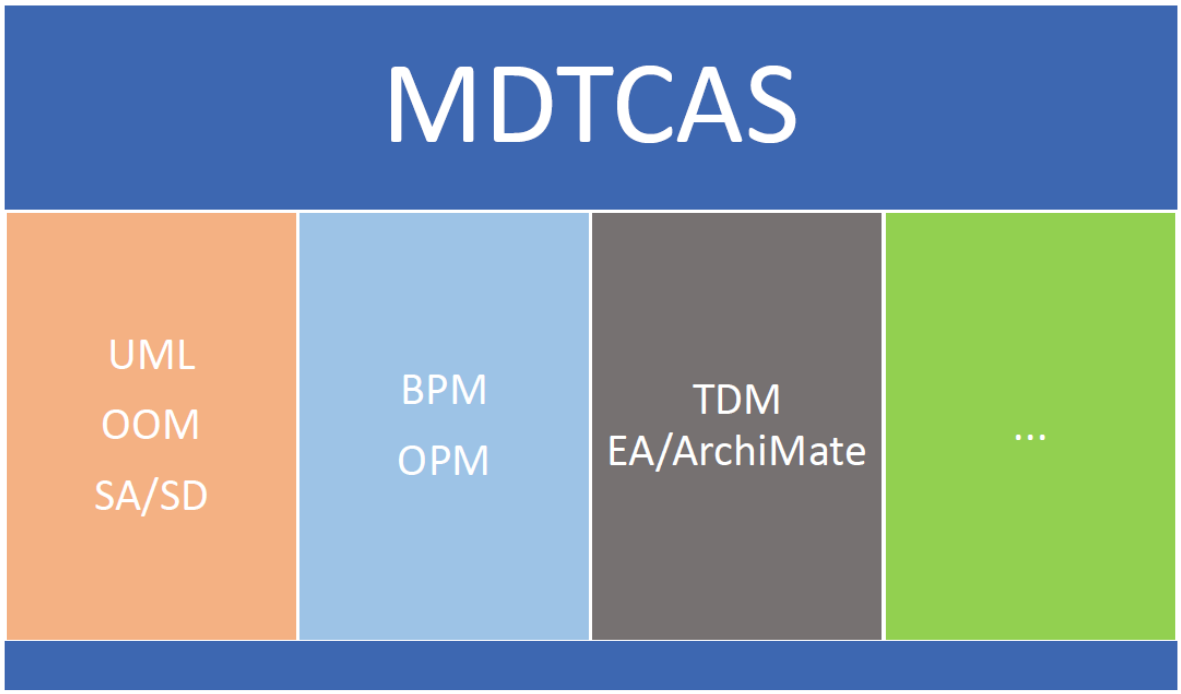


Figure 3. MDTCAS' Implementation.

The key to linking these two architectural domains are BPs, OPMs, and Business Process Models (BPM) which are subsets of process architecture(s). Polymathic-holistic overview/visibility across all APD's CSAs, helps *Project Managers* and teams, to predict the *butterfly effects* (how actions can have huge effects on the course of a major event) [7]. Where OPMs and BPMs are incorporated in BBs and CBBs, which will be presented in PoC's Phases 1 and 2.

Phases 1 and 2

As shown in Figure 4, Phase 1 contains the literature review, PPMC Tables (CSAs) evaluations, and delivers the decision to continue to (or not) Phase 2. The literature review's outcome supports PoC's background, using an archive of an important set of references and links that are analysed using a specific interface. After selecting the CSAs/CSFs tags are linked to various STORM BBs/Microartefacts scenarios; and this concludes Phase 1. The DMS4PPMC -related PoC (or Phase 2), uses the HDT to deliver possible solutions. The empirical part is based on the AHMMPPMC's instance and STORM's Microartefacts mechanics, which uses the internal initial sets of CSFs that are used in phases 1 and 2. The *Project's* enumeration of CSAs are: 1) PPMC's Appliance; 2) ICS and services; 3) Enterprise patterns integration; 4) EA for STORM; and 5) STORM based DMS4PPMC. Tables 1 to 5 were presented and evaluated in this article and they are this article's empirical part. The Tables processing was influenced by the Object Management Group's (OMG) Decision Model and Notation (DMN), where the DMN can be used for the specification of business decisions and business rules. DMN is optimal for initial checking based on decision making [58]. STORM delivers recommendations on how to use it with an IHI framework.

Figure 4. Phases 1 and 2 flow.

PPMC'S Appliance CSFs

Based on the AHMM4PPMC, LRP4PPMC and DMS4PPMC, this CSA's CSFs/KPI were weighted by the HDT.eval function and the results are shown in Table 1.

Critical Success Factors	KPIs	Weightings
CSF_PPMC_Appliance_Polymathic_Approach	Proven	From 1 to 10. **10 Selected**
CSF_PPMC_Appliance_Factors_Integration	Proven	From 1 to 10. **10 Selected**
CSF_PPMC_Appliance_RP_Integration	Complex	From 1 to 10. **08 Selected**
CSF_PPMC_Appliance_AHMM4PPMC_STORM/SWOT	Feasible	From 1 to 10. **09 Selected**
CSF_PPMC_Appliance_Linking_STORM/SWOT_Factors	Feasible	From 1 to 10. **09 Selected**
CSF_PPMC_Appliance_ModelFirstApproach	Feasible	From 1 to 10. **09 Selected**
CSF_PPMC_Appliance_IHI_Project	Possible	From 1 to 10. **09 Selected**
CSF_PPMC_Appliance_Phase_1_2	Proven	From 1 to 10. **10 Selected**

valuation

Table 1. This CSA has the average of 9.25.

This CSA's result of 9.25, which is high, is mainly because the iteratively used RDP4PPMC is mature and that the RP to deliver basic *Artefacts* was successful. But that does mean that the RP and PPMC's are feasible. As this CSA presented positive results, the next CSA to be analysed is ICS and service. The *Project*, PPMC, and STORM/SWOT depends on the role and the status of the *Entity*'s ICS and generated services.

ICS AND SERVICES

The Roles of Standards, Avant-garde Technologies, and Methodologies

Today there are many APD/business, EA, services, and ICS related standards and they are to some degree applicable, like the following ones: TOGAF (and its ADM), SOA, CMMi, COBIT, ITIL, UML, BPMN, BMM, SysML, SOA/MSA,… These standards, methodologies, services technologies, and their modelling/implementation environments, support the breakdown unbundling of legacy ICS systems, by the use of empirical and iterative approaches, which can be supported by DevOps. An important goal in Projects, can be: *The changes done on the traditional ICS of Entities, to become agile innovative ones, should be*

based on STORM and PPMC…[8]. The integration of related Microartefacts can be used by adopted standards. The PPMC englobes templates for the use of *Project's*Architecture Building Blocks (ABB) and SBBs. The theory and concept of reusable BBs or patterns. The PPMC, suggests that implementers of servicespatterns must be able to reuse proven components that emerge from the best *architecture & modelling* practices, to solve genericPIP requests. Without the use of PPMC, ABBs, SBBs and patterns, projects would not be applying *architecture & modelling* techniques, and that results in, that the *targeted business solution*: 1) Has bad performance; 2) Lacks scalabilities; 3) Brings human instabilities; and to 4) Becomes un-usable and un-maintainable. Added to that, for practical reasons, many EAand/or ICSspecialists have the tendency to *reinvent the wheel*, when attempting to implement project templates. Therefore, the PPMC must apply: 1) Standardized tools and frameworks, like TOGAF and/or UML; 2) Standardized services' modelling methodology, like SOA Markup Language (SOAML); 3) Standardized BPM; 4) Apply a mapping model; 5) Apply STORM; and 6) Use the optimal agile concept for the PPMC.

Agility Concepts for the PPMC

*Project's*agility is achieved by combining various domains, like: STROM feedbacks/results, Synchronized APD/business engineering concepts, ICS,and PPMC/EA related methodologies that promote global APD's automation schema to be implemented in various levels of *Entity's* ICS. In order to, unbundle and maintain the existing legacy ICS and glue its innovated/generatedBBs/Microartefacts mapping links in its dynamic and transformed ICS modules. ICS modules are made up of BBs/Microartefacts, where each BBs/Microartefact is a set of micro (business) services which can be: 1) SOA based services; 2) Microservices; 3) REpresentational State Transfer (REST) services; 4) APIs abstracted services; 5) Interfaced legacy modules...There is nofinal definition of (business) services architectural style, but there are common characteristics around the *Entity*, APD/business capability, Artificial Intelligence (AI)/business intelligence, and decentralized control of business environments. The transformed agile business system becomes coherently automated by unbundling of the legacy ICS. This unbundling process delivers the needed sets of SBBs, where and ABB is a set of abstracted services' models. This process starts with the classification of services/SBBs (or Microartefacts) into CSDs. SBBs (or services) can be interfaced by using the API approach that is based on [9]: 1) Modelling API's schema by creating a design document; 2) A schema model is a contract between the *Entity* and the clients; 3) A schema model is essentially a contract describing what the API is and how it works; 4) It facilitates STORM activities,and 5) Uses an agile strategy. The RP based unbundling process is setup by the *Project's* team who synchronize them with sets of requirements, related SBBs, by using implementation and tests procedures.

Implementation, Tests and Tools Diversity

The Behaviour-Driven Development (BDD) extends existing unit testing with automated acceptance testing. A BDD script enables clear communication and continuous interaction between APD, development and testing specialists. ExistingUnit Testing (UT) is still the foundation of automated testing and is required for efficient BBs, CBBs and Microartefacts verifications. UTsfocuson software code classes' activities (such as statement, branch, and path coverage), whereas BDD ensures software quality from anAPD/business-oriented viewpoint and can be used to test STORM scenarios/scripts. UTs have the highest number of test cases and coverage.Using many tools or IDE and gadgets, can generate*Project* problems. It is assumed that IDEs and associated tools can solve all types of *Project* problems. Instead of using straightforward EA modelling/PIP,engineers spend most of their efforts in the search for libraries, scripts or gadgets which would shorten PIP time. Therefore, the focus must be set on: 1) EA, SBBs, Microartefacts modelling; 2) Unit, aggregated and integration tests; 3)

Agile and continuous integration and deployment; 4) Change management concepts; 5) Performance and robustness estimations; 6) Implementing Factors and especially KPIs in SBBs; 6) Choosing random risk methods, where STORM can give a clear view; 7) and many others… BDD enhances the simplistic Test-Driven Development (TDD) approach by integrating Behavioural aspects, Features, … To be used for testing and then executingCBBs, BBs and Microartefacts. That all depends on the level of granularity and the status of the RP based unbundling process.

Granularity, Mapping and Unbundling

Defining PPMC's, mapping and BBs/Microartefacts' granularities for a*Project* is a complex undertaking, added to that the "1:1" mapping and classification concept is a long process; but it is crucial one. Mapping of a requirement's UC(s) to BBs/Microartefact(s)/services in the form of a class diagram or communication diagram, can be done usingArchiMate or UML/OOM.

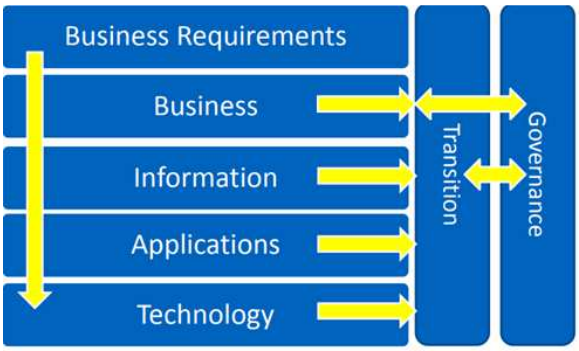


Figure 5. The *Project's* interaction with various ICS modules.

This modelling and mapping concept is supported by a set of Microartefacts where its DMS4PPMC can evaluate STORM's status. DevOps can use the PPMC/DMS4PPMC to evaluate the requirements, services' integration to deliver strategy's solutions. A requirement and Microartefact map to a class diagram (or communication diagram) and has a Global Unique IDentifier (GUID). PPMC's unit of work or a project Microartefact, is based on the alignment and classification of all the *Entity*'s requirements (and resources). project agility is achieved by combining synchronized domain, ICS and PIP methodologies that promote *Entity's* automation and business robustness. To unbundle, restructure and maintain the existing ICS and to glue its innovated Microartefacts in its choreography modules, and the DevOps process, at various EA levels as shown in Figure 5. The DevOps contains automated script to manage Microartefacts by applying a set of actions that coordinate and control PIP activities. An ADM managed DevOps process is based on a holistic systemic approach and its mechanics manage Microartefacts/services when it receivesPPMC, STORM or other change requests. DevOps interacts with a multitude of projectmembers, components, and resources, in a synchronized manner.The ADM assistsPPMC's integration activities [8], in which DevOps supports mapping mechanisms that use the PPMC/DMS to make the*Project's*integration flexible and to avoid and solve major problems, which can be facilitated using Cloud computing.

The Role of Cloud Computing

The Cloud based Compute-Systems' (CbCS) support *Projects*with its virtual secured ICSand contains an integrated empiric DMS. The CbCS needs a Cloud infrastructure that is supported by the alignment of various existing Cloud Platform (CP) standards, EA paradigms, and different PIP strategies, where an important goal is to support AI capacities. The Google CP (GCP) was chosen as a sample CP to prove this article's feasibility and the possible alignment

to standards, but the CbCS can be applied to any type of Cloud. There is need to design and implement a standardized commercial CP architecture or methodology, and corresponding procedures, because the organization can build its own private Cloud solution and these facts protect the organization from being locked-in. The CbCS is based on research resources related to Cloud CSs and Cloud security, to offer a set of PPMC recommendations, which can be applied to enable Cloud based ICSs. The CbCS strategy is a generic Cloud-driven approach that uses EA and any type of CP, like the GCP. The GCP includes various CS resources that offer different levels of controls, features, and ICS management and design support. CS resources need different levels of provisioning, that depend on the used CS service. CS topics include: 1) Use of preemptible and standard Virtual Machines (VM) in Compute Engines (CE); 2) App Engine (AE) in two forms: Standard (AES) and AE Flexible (AEF); 3) Design of Kubernetes clusters; and 4) Deploying Cloud Functions (CF). CSs use the Infrastructure-as-Code (IaC) for network configuration and infrastructure provisioning. The CbCS in the case of the GCP, includes the following activities and components: 1) Designing CSs; 2) Relating CSs and Use Cases (UC); 3) CE's integration; 4) AE's integration; 5) Kubernetes Engine (KE) integration; 6) CF's usage; 7) CS provisioning; 8) Security and advanced design issues; 9) Managing states in distributed CSs; 10) Data flows and pipelines; and 11) Monitoring and alerting. CE is Google's Infrastructure as a Service (IaaS) concept and the core functionality provided by CE is VMs. AE is a Platform as a Service (PaaS) concept, where AE users do not have to configure servers, but they use applications that run in AEs; where there are two types of AE: 1) AES; and 2) AEF. KE is a managed service offering cluster management and container orchestration. KE allocates cluster resources, manages containers, performs health checks, and manages VM lifecycles using CE's instance groups.

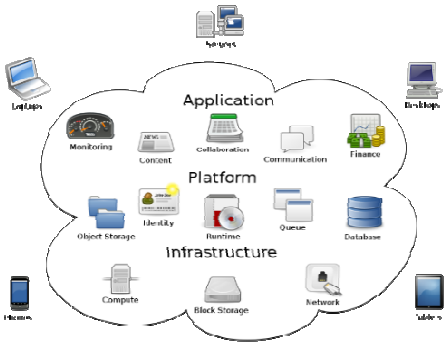


Figure 6. A generic CP [12].

CFs is a serverless compute service for event processing, and it is designed to execute code in response to events. Other CbCS aspects when designing the platform, are managing state in distributed systems, data flows, and monitoring and alerting; and above all the just-in-time AI requests. As shown in Figure 6, CP includes a group of networked components providing services, which do not need to be individually located. The CbCS provides an entire managed suite of ICS platform components, which can be IHI environment. *Entity*'s Private CP (PCP), enables the processing of its APD/business activities, which include a large set of BBs/Microartefacts, applications and resources. The *Entity's* set of applications and resources are managed by the PCP, where applications are used to serve internal or end clients. The DMS supports the CbCS to serve*Entity*'s APD/business capabilities in operating in various fields, like business data management, security management, business services, policy making, regulatory and governance activities. The CbCS uses central domains, like EA, CP,

and services' coordination; where a PCP can build on existing patterns and technologies like APIs [10, 11, 12, 13].

The API's Usage

The REST concept is based on Create Read Update Delete (CRUD) operations, which embed thefollowing operations: 1) POST, which lists, paginates, filters the lists of attributes of object(s); 2) GET, which retrieves the representation of an object; 3) PATCH which updates specific attributes of an object; and 4) DELETE which deletes a specific attribute from an object.

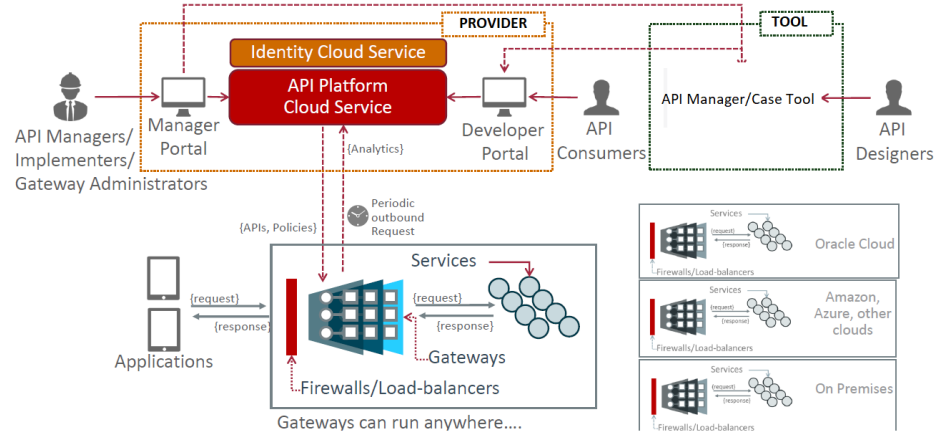


Figure 7. API Platform-Provider [14, 16].

*Entities*can use STORM to minimize risks, and for that they need to ensure their APIs are multi-tier resilient, persistent, high performance, controlled, robust, and secure. An API has a complex flow as shown in Figure 7. API gateway is the bridge that is used to access stored other APIs and abstracted services. The PPMC manages flows between API clients/interfaces and the server SBBs that exposes APD's model. Such a bridge (coordinated interface) handles security issues like authentication and authorization, request routing to backends, rate limiting, to avoid system's bottlenecks and to protect against security attacks, and to handle various types errors or model's exceptions [14, 15, 16].

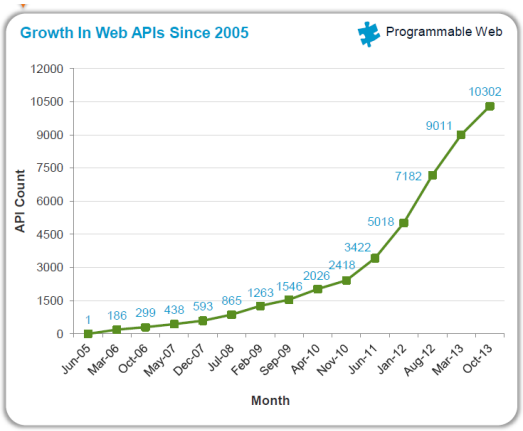


Figure 8. API's exponential growth [14, 15].

In general, API management refers to the process of managing APIs' calls through their full DevOps lifecycle, including defining, deploying and publishing them, monitoring their performance, and analysing usage patterns to maximize business value and ICS's robustness

[14, 15, 16]. The API Pattern (APIP) is a central pattern taking in account their fulgurant growth as shown in Figure 8.

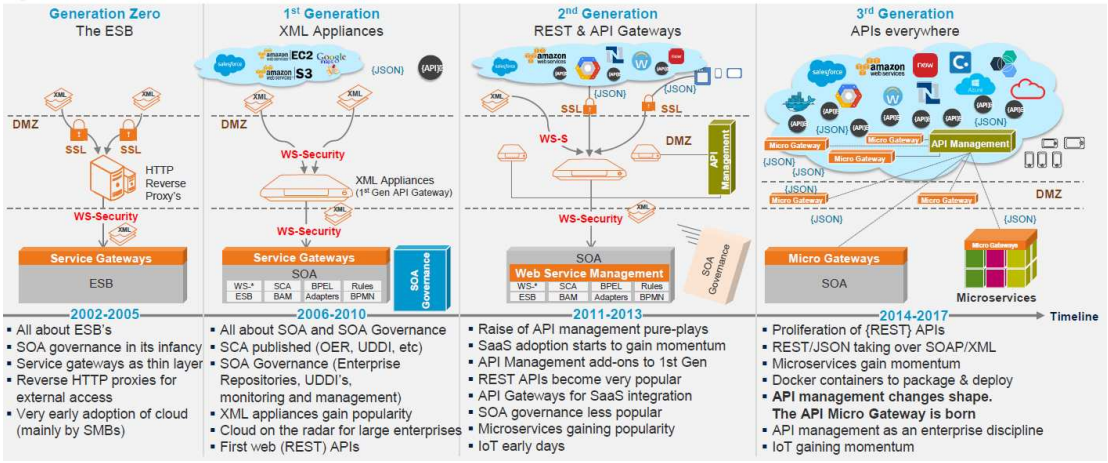


Figure 9. API-From Generation Zero to 3rd Generation API Management [14, 15]

As shown in Figure 8, it is interesting to view the various phases of evolution of perspectives on API Management; and to notice the evolution of API's Management and API's space. Such a perspective on how the space and related practices have evolved, allows *Projects* to improve API Management and the use of STORM.As shown in Figure 9, this is the most common API Gateway pattern, and it follows the traditional Application Delivery Controller (ADC) architecture. In this pattern, the gateway handles all types of activities like[16]: SSL/TLS'Termination, Authentication, Authorization, Request routing, Rate limiting, Request/Response manipulation, and Façade routing.

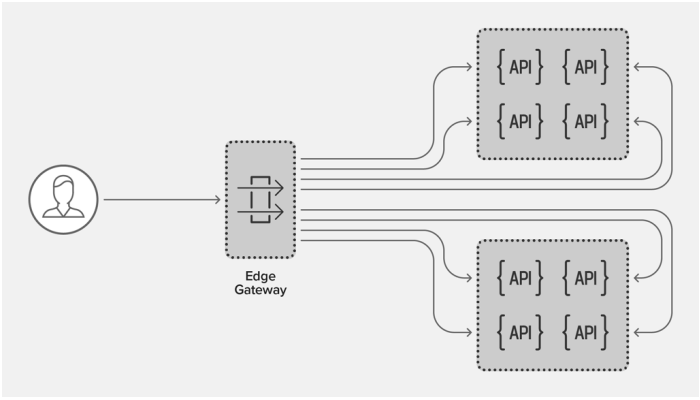


Figure 10. Gateway for Services [16]

The API Gateway approach is optimal for publicly exposing SBBs from monolithic applications with centralized governance. But it is not well□suited for MSA or situations that require frequent and profound changes. Traditional interface gateways are optimized for north□south traffic and are not able to efficiently handle huge volumes of east□west traffic generated in distributed MSA based ICS, as shown in Figure 10 [16]. That is why it is recommended to englobe various pattern sets and their relationships in BBs, by using the Enterprise Service Bus (ESB)Patterns (ESBP).

ICS and Services' CSFs

Based on the AHMM4PPMC, LRP4PPMC and DMS4PPMC, this CSA's CSFs/KPI were weighted by the HDT.eval function and the results are shown in Table 2. This CSA's result of 8.30, which is low and insufficient. This is mainly due to the fact that integrating ICS and

services is complex. As this CSA presented positive results, the next CSA to be analysed is enterprise patterns integration that support STORM.

Critical Success Factors	AHMM4CBB enhances: KPIs	Weightings
CSF_ICS_Sevices_Standards_Technologies_Methodologies_	Complex	From 1 to 10. **08 Selected**
CSF_ICS_Sevices_Agility_Concepts_	Complex	From 1 to 10. **08 Selected**
CSF_ICS_Sevices_Implementation_Tests_	Feasible	From 1 to 10. **09 Selected**
CSF_ICS_Sevices_Granularity_Mapping_	Complex	From 1 to 10. **08 Selected**
CSF_ICS_Sevices_Cloud_Computing_Integration_	Complex	From 1 to 10. **08 Selected**
CSF_ICS_Sevices_API_Usage_	Feasible	From 1 to 10. **09 Selected**

valuation

Table 2. This CSA has the average of 8.30

ENTERPRISE PATTERNS INTEGRATION

EA's Integration Construct

The PPMC provides a concept for classifying and using existing types of patterns, BBs, Microartefacts(simply *Artefacts*) in anEA IntegrationConstruct (EAIC), that can support *Projects*and STORM's usage. The EAIC is based on a composite *Artefacts* model, which can be used as a template to instantiate BBs to implement a variety of types of *Projects*. Increasingly complex, competitive, and automated APD environments like mechanistic *Entities* are the essence for investment in dynamic EAs and the transformation of flexible and efficient APD environments. An*Artefact* has already been defined as: …*an idea that has been useful in one practical context and will probably be useful in others*" [17, 18].In standard EA methodologies like TOGAF, patterns are to be considered as a concept for using *Artefacts* in *Project's* context; like in the case of a re-usable solution to a *Project* problem. The use of patterns may support EA practitioners to identify combinations of ABBs and/or SBBs, which have been implemented and verified to deliver successful solutions and support STORM.A successful finalization of the implementation phase can give an important business advantage and can guarantee the transformed *Entity's* perennity. EA methodologies like TOGAF and its ADM, manage the layers [19]: 1) Business; 2) Information System, and 3) Technology. All these layers map to ArchiMate layers, as shown in Figure 11, which supports *Project's* Vision and can be verified with STORM.

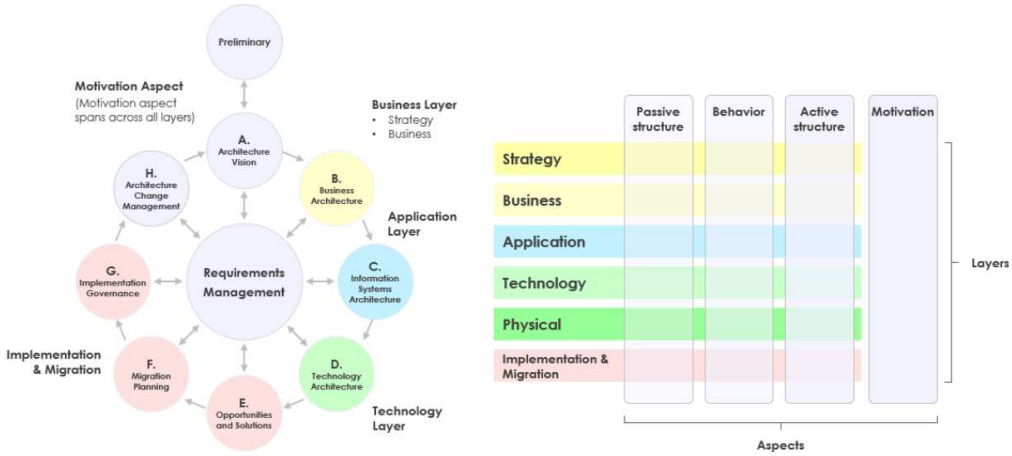


Figure 11. ADM's Phases mapping to ArchiMate layers [19]

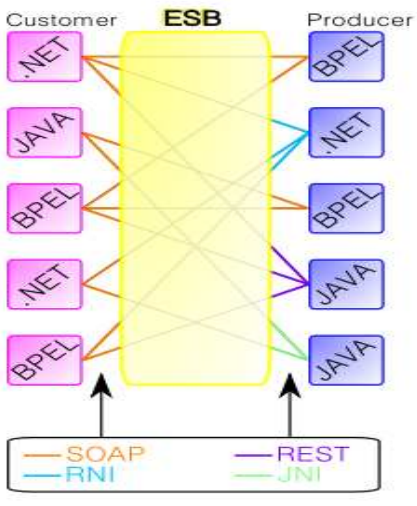
Extracting Artefacts-Patterns for BBs and CBBs

Before introducing how EAIC driven STORM are implemented, it is important to prepare the *Entity*is ready for using the basic *Artefacts*, like patterns, CBBs/BBs, classes, composite objects, tables or services to transform the ICS by applying the RPto extract Patterns. The RP will support the agile and autonomic *Project's*PIP to implement pattern instances. That needs a precise EA mapping concept that must use a standard framework [20]; that is the main *Project* principle. The RP supports an agile iterative model that can map all *Project's* artefacts in a linear 1:1 manner. The RP tries to extract granular/atomic classes that are needed for the future BBs-based patterns; so that each *Project* artefact can be managed independently. This applies a structured unbundling process, by using the 1:1 mapping rules that are based on services, or composite classes. The various types of *Artefacts*:

- *Generic Patterns*, where generics in ICS related domains are the common basics for implementing BBs; in some domains Generic Patterns (GP) are known as templates (known as polymorphism). Templates facilitate the phases of design and implementation of STORMand other types of components. The most known GPs are: 1) Singleton; 2) Linked List; and 3) Visitor. GP's main PPMC are to extend work elements and to preserve the level of abstraction [21]. This is the minimal set to be empowered with other *Project* specialized basic patterns to support logging, security, data-management… The *Projects* denotes the basic patterns category as the enterprise's basic implementation patterns. The evolution of patterns made it possible to create *Architecture and Design Patterns* that are the predecessors of ABBs.
- *Architecture and Design Patterns* are used in software architecture, design, or implementation *Project* phases. In *Pattern-Oriented Software Architecture*, a *System of Patterns* can have the following three types of patterns [22]: 1) An Architecture Pattern; 2) A Design Pattern; and 3) An Idiom. EAIC will try to find analogous concepts and terminology, and offer a re-usable holistic pattern, that is a composite model of Design Patterns.BBs represent implementation's best practices that can be used by a project team of experienced object-oriented implementers. BBs are solutions to generic problems that implementers can use for solving standard and recurrent problems which are faced during the project's PIP. Types of BBs: 1) Creational patterns, are a set of design patterns PPMC with common composite constructs that can be used in a project**,** where their main activity is the instantiation of BBs or services. Creational patterns support a standardized mechanism to factorize the end system's BBs or services; 2) Structural patternsmanage BBs by delegating their behaviour to other BBs, what permits the creation of a layered architecture of components, using loose coupling, facilitating BB's communication and accessibility. This PPMC pattern provides manners to structure a composite BB so that it can be instantiated in using minimum end system's resources; and 3) Behaviouralpatterns, focuse on BB's algorithms, and itsPPMC focuses on the communication among? project's artefacts. EAIC will try to find analogous concepts and terminology, and offer a re-usable holistic pattern, that is a composite model of Enterprise Patterns.
- *Enterprise Patterns and Enterprise Architecture Patterns*, in which the Model View Controller (MVC) pattern is the most important and it offers interfaces for messaging and a related data model that serves as a messaging framework, used as an integration server. The messaging framework is essential for complex system integration [17, 18, 23]. Enterprise integration patterns are the base for building the EA patterns. EApatterns, manage: 1) concurrent access to databases; 2) applications' user interface to applications; and 3) transformations of legacy system. The EA Pattern (EAP) set includes the: 1) Domain Logic Patterns; 2) Data Source Architectural Patterns; 3) Object Relational Behavioural Patterns; 4) Object-Relational Structural Patterns; 5)

Object-Relational Metadata Mapping Patterns; 6) Web Presentation Patterns; 7) Distribution Patterns; 8) Offline Concurrency Patterns; 9) Session State Patterns, and 10) Base Patterns. A set of EA patterns serve to build an enterprise pattern that can serve as Common Denominator Patterns (CDP) for or reference model(s).

- *Model View Control Pattern*,is acentral pattern andalso is the most complex and complete pattern that must be analysed separately and considered a category. The MVC decouples the: 1) Modelling of data and the domain CDPs; 2) Presentation CDP, and 3) Actions or services thar are based on Graphical User Interface (GUI) [27].
- The *EAP* contains: 1) Basics, used to define Architecture Pattern (ARCP), which is a fundamental architecture element to support ICS' transformation vision.That should be crafted in an applicable *Project* framework or concept. This IHI framework should include easy to integrate patterns, andcan change ICS's architecture and its implementation outcomes. The ICS resources can be used in EAIC, which can be applied to support services of crucial importance for the PIP; where these patterns can be adapted in a just-in-time manner, by using services; 2) Management,by using thePPMCto govern or control the ICS resource patterns for STORM. Unfortunately, adaptable ICS resource patterns for such undertakings are still in an infancy age or have a hermetic approach like the MSA. An ICS resource pattern can be used in the *Entity's* production activities, which comes after the finalization of the PIP, to control and govern the resultant APD/business system. The ICS resource's pattern main component is the service that manages the implementation of services. In this article the authors present a set of ICS resource recommendations in the form of reusable patterns to promote the optimal EA models. As SOA is fundamental for TOGAF, ADM and other disciplines, the SOA Pattern (SARP)is used for basic services operations.
- The SARP is a design patterns catalogue (published by Arcitura Education) supports SOA standards. These patterns encompass service-oriented architecture and service technology (Arcitura, 2020): Foundational Inventory Patterns: Canonical Protocol, Logical Inventory Layer Patterns, Inventory Centralization Patterns, Inventory Implementation Patterns, Inventory Governance Patterns, Foundational Service Patterns, Service Implementation Patterns, Service Security Patterns, Service Contract Design Patterns, Service Governance Patterns, Capability Composition Patterns, Service Messaging Patterns, Composition Implementation Patterns, Service Interaction Security Patterns, Transformation Patterns, REST inspired Patterns, Composite Patterns.
- The BPM Patterns (BPMP) are patterns that show how to model and connect activities together, in order to solve a *Project Problem*. BPs are like motorways, as we drive, we become used to similar and time proven motorways. Countries ensure that their engineers follow proven specifications therefore motorway constructions are consistent. BPMP are the specifications of motorways of BPs [28]: Basic Control Patterns, Advanced Branching and Synchronization Patterns, Structural Patterns, Multiple Instance Patterns, State Based Patterns, Cancellation Patterns: Cancel Activity Pattern. That is why Pattern and Cancel Case it is recommended to englobe various pattern sets, including RESP.



The API/ REST and ESB Patterns

Figure 12. The ESB integration with development environments [29]

The CRUD based REST pattern embeds needed APIs which are managed by an ESB. The ESBs support the integration of all the mentioned technology standards. Standardized *Project*s must be transparent regarding their solutions and their focus must be on their business engineering choreography, regardless of the business domain. BTM's integrated enterprise patterns are using the following methodologies: 1) TOGAF's ADM that adopts the UML's spiral model; and 2) project management concepts. Once the *Project*'s standards are established, a pre-enterprise patterns architecture blueprint must be defined. If the unbundling process is successful, the *Project* maps all BBs to services and BBs. These BBs or services can be called via the enterprise service bus, as shown in Figure 12. The *Project* identifies the set of main patterns and related patterns to implement the EAIC, which addresses these various topics and delivers a common concept.There are many of them, but the authors will take the most important ones to present this article's background. The various types of patterns addressed by the EAIC are:

- Generic Patterns (GENP)
- GoF Patterns (GOFP).
- BPM Patterns (BPMP).
- API Patterns (APIP).
- SOA Patterns (SOAP).
- ESBPatterns (ESBP).
- Enterprise Applications Integration (EAI) Patterns (EAIP).
- Cloud Computing Design (CCD) Patterns (CCDP).
- Organizational (ORG) Patterns (ORGP).
- Architecture Patterns (ARCP).
- REST Patterns (RESP).
- MicroServices Patterns (MSRP).
- Messaging Patterns (MSGP).
- In House Composite Patterns (IHCP).
- And many others…

Because of this article's limitations SOA and BPMP will be analyzed. The *Project's* classify and store patterns and other *Artefacts*are? in the *Entity's* Continuum.

Continuum, Reference Models, and the Enterprise Meta-Model

TOGAF's technical reference model offers an interface to manage entries, like enterprise patterns. Enterprise patterns are mainly a part of the modelling component. To integrate enterprise patterns, the *Project* must use existing standards and methodologies [24]. EAIC will try to find analogous concepts and terminology, and offer a re-usable holistic pattern, that is a composite model of related patterns.Patterns' relationships use OO relationship types. Relationships interconnect patterns, creating a virtual composite pattern (or EAIC), indicating how the DMS solves types of problems. Related patterns to implement the EAIC need a well-designed *Entity*Meta Model (EMM).A Virtual Meta Model (VMM) is UML's expression of a formal model with a defined set of UML, ArchiMate or other extensions. The EMM is a VMM variant and uses a modelling language [25].The Design Pattern Modelling Language (DPML) is a notation that supports the specification of BB solutions (or ABBs) and their instantiation into UML or ArchiMate models (or SBBs). DPML provides constructs which allow BB solutions to be modelled and integrated. BBs are described using a mixture of natural language and UML style diagrams, this causes complex scenarios in integrating BBs in the ICS. As shown in Figure 6, a DPML models BB to support ABBs, SBBs, services, SBBs and CDPs [26]. The EMM is an ontology for EA concepts using EA frameworks and tools. EMM's intention is to provide extensible set of concepts and corresponding relationships, with semantics that can be mapped to the patterns (or anEAIC), ABBs, concepts, activities, and standard case tools of frameworks. The EMM can map to existing frameworks, like the Ministry of Defence Architectural Framework (MoDAF), the Federal Enterprise Architecture Framework (FEAF), TOGAF, and other… EMM is applied to abstract the complexity of these frameworks and the EAIC in a *Project*. The abstraction views are: Conceptual, Logical, and Physical. The core of the EMM has layers representing the areas of EA to be included and rows representing the levels of abstraction, or views. Although ARCPs have not been integrated in standard EA methodologies, such as TOGAF, where in its first four main ADM Phases (Phases A to D), it gives a clear indication which resources should be used. There are used re-usable resources, like EAIC, which is managed by the EA Continuum, as shown in Figure 9 and contains the major CDP. An *Entity* that adopts a formal approach to apply ARCPs, must integrate them in their *Entity's* Continuum, to support various Views and the ADM, to support the *Entity's* composite construct.

The Entity's Composite Construct

This article's goal is not to present one more time the various types of patterns, but to show the optimal manner to do the preparations needed to integrate patterns in a coherent *Project* architecture that would use PPMC and STORM by using:

- The implementation patterns: These types of patterns understand a family of patterns that have a composite structure, like 1) the classical design patterns and 2) the services patterns.
- The integration and enterprise patterns: These types of patterns understand patterns that have a component structure, like 1) Integration patterns; 2) the EA patterns; and 3) the BPMP.
- The dynamic EMM as shown in Figure 13, is continuously transformed by the RP.

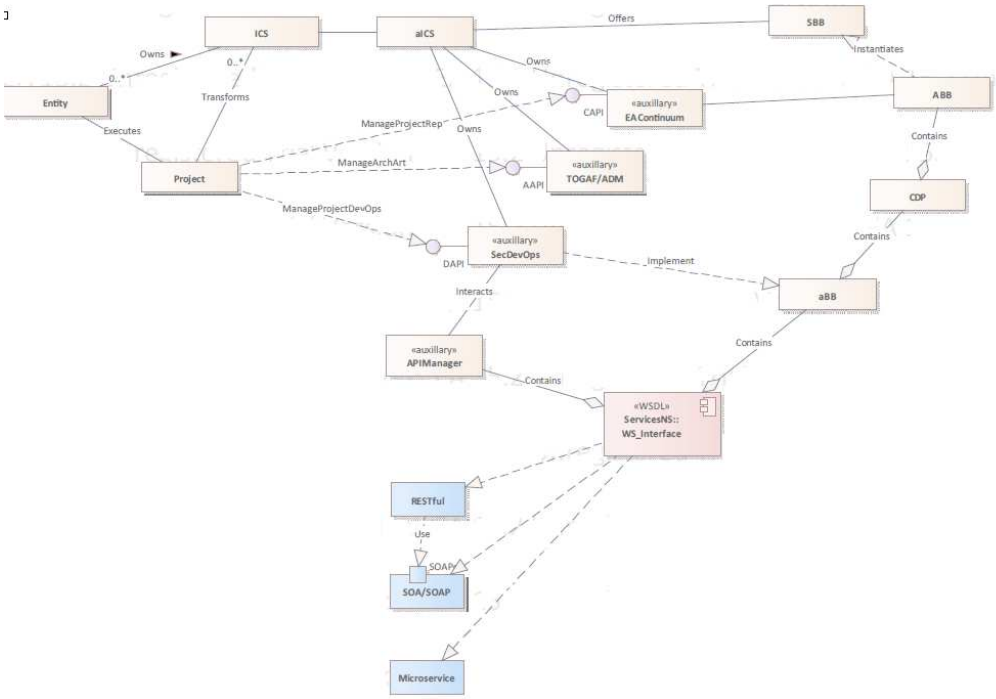


Figure 13. The EMM in the defined context

The Transformational RP

The transformational RP can use some of existing transformational concepts or patterns like the *Strangler Pattern*(SP) that is used for MSA based systems. Instead of the very risky transform of all, the best alternative is to use an agile approach based on the EAIC and evaluated by STORM. Such an approach recommends observing the current *Artefacts* component.Traditional componentsare iteratively replacedby transformed ones. This approach offers an evolutive EAIC-based plan that PPMC controls for possible problems, by reducing the risk that is associated with a brutal change; and STORM evaluates the strategy. ItsPPMC offers value back to the business by enabling a fast delivery of transformed features, until the transformed componentsare mature and can replace the legacy one. Such an approach is well-known; in fact, in 2014, it was hammered and designed by Chris Stevenson and Andy Pols. Martin Fowler, who promoted the SP,gave it a name: *The Strangler Application;* which is in fact a pattern. The basic idea is to break the legacy monolith into smaller parts (*Artefacts* in which *Factors* can be integrated and then evaluated by STORM), which requires precise preparations and a concise concept. The concept must support a smooth transformation process and the sustainable operation of the business, and at the same time it processes incoming requirements. That needs that an *Entity* has the capability to transform legacy components, by [31]:

- Extracting code blocks: Frequently, in this manner, which is based on copying software modulesto createnew components, is the default mode. This option can reintroduce bugs from the old system. It is PPMC susceptible to the RP based transformation effect, in which developers placed high value in the former versions, which had not been implemented by the same engineers, and where for the reuse of code it has to be rewritten and refactored.
- Rewriting the capability: Initially, this manner can be considered as an expensive route compared to copying code, but the benefits of rewriting, by capability offer the optimal Return on Investment (ROI), which can be confirmed by STORM. When rewriting, the EAICdelivery teams can question

legacy assumptions, revisit the business problem, and improve the business process by an optimal approach. The code parts in legacy systems may not have been built on concise domain concepts or even obvious separation of concerns. The rewriting process offers a chance to correct and optimize the code parts.

In many *Entities*, a unit of software/code is responsible for specific operations.This unit of code can be a sensitive piece of the component which might be seen as a good one for extraction and possible reuse. But in the case of rewriting, the EAICteam can revisit the process and consider the STORM mitigated strategic objectives of the *Entity*. The EAICteam might replace the actual legacy part with an up-to-date architecture or even choose an external component. That enables new forms of activities for the *Entity*. If the legacy code is simple and performs a basic operation, has clear domain concepts, or had an important intellectual property value, in such a case, like in algorithms, most EAICteams would opt just rewriting this part's capability. Breaking down a legacy monolith, requires a precise STORM based EAICstrategy, which will guide the *Project* team, the implementation engineers, the team of architects, the business analysts, and other team members. This a very difficult process, because legacy monoliths have, in general no proper separation of concerns, no concise domain design, and in some cases, many technological weaknesses. By transforming the legacy monolith in terms of capability, it is possible to accomplish *business-value-oriented* implementation by prioritizing the requirements to extract resources, by using SP, in such a way that it adds value to the business, and it balances important risks by using STORM. For that goal, there are various methods and strategies to gradually move the usage and to lift the capabilities and functions to the SP-based application, by applying [31]:

- Event tapping is known as an event interception mechanism.In this strategy, whenever there are event-driven components or capabilities, there is the possibility to tap in to the stream of events; and then start to build or replace call-back functions for those events. This mechanism/pattern allows for building a parallel system for ensuring business continuity.
- Asset capture: In this case, every component manages a set of functional objects or assets, like user accounts, transactions, historical records, or product orders. Transforming the capabilities of managing these assets independently and using them in SP-based application. Such anRP based transformation is more of an art than a science, but in general, this strategy provides for a path to create services with clear domain concepts and responsibilities.
- Service bubbles: Practically all the applications and components are artefacts that consume a set of APIs. Tay, most ICSs accept service-oriented concepts, from design to delivery. Therefore, the *Project* team must explore chunking and refactoring the legacy monolith into a service-oriented concept.
- Apply STORM to re-evaluate the strategy and risks related to SP.

The SP-based strategy examines the *Entity*'s capabilities to transform the legacy system, and to create small strangler-based services that encapsulate the logic of each capability in independent *Artefacts*. Such an approach eventually leads to the creation of mesh services, which support the entire set of capabilities of the previous legacy system. Finally, that would allow PPMCto manageEAICservices-based architecture, which is done by splitting software units into independent services that are organized around a business capability which is evaluated by STORM. By using such a strategy, the ***Project*** team can chip away the capabilities of the legacy system by migrating business capabilities into independent services.

The business value that is delivered is due to the prioritization and a proper mix of transformed logic and new requirements delivering the ROI [31]. The SP supports the integration services-based*Artefacts* and BPMs.

Services-based Artefactsand BPMs

Today, there are many types of services, like standard functions, business services, SOA/SOAP, REST, MSA,… In order to simplify the services integration in the 1:1 concept, the authors propose a service which has the following capabilities [34,35]: 1) To facilitate the PIP and the usage of standards; 2) To manage and assemble *Project'sArtefacts*; 3) To make services agile, reusable and easily replaceable; 4) It can be implemented in many SBBs; 5) It has a GUID; 6) It asserts the *Project's* 1:1 mapping concept;6) It enables business activities' interoperability and integration; and 7) It uses PPMC and STORM control and evaluates services-based *Artefacts* and BPMs integration. SARP embeds the SBB pattern to offer a simple interface which can be managed by a business specialist. SBBs must be adequately classified and interconnected using interoperability standards [28]. SARP is one of the most important and complete set of patterns; analysing the number and content of these patterns, which shows the complexity of usage and integration of various sets of patterns by using the EAIC.

The EAIC

Fundaments

EAIC can be used to abstract EA artefacts and *Artefacts* for *Projects***,** where *Project* engineers implement interconnected patterns. The complexity, diversity and cross-functional nature of EA and other *Project*'s domains require that various categories of patterns should be developed and classified in various disciplines, domains, and levels of precision. The integration of various categories of patterns and their non-standardization cause this topic's lack of maturity [30].

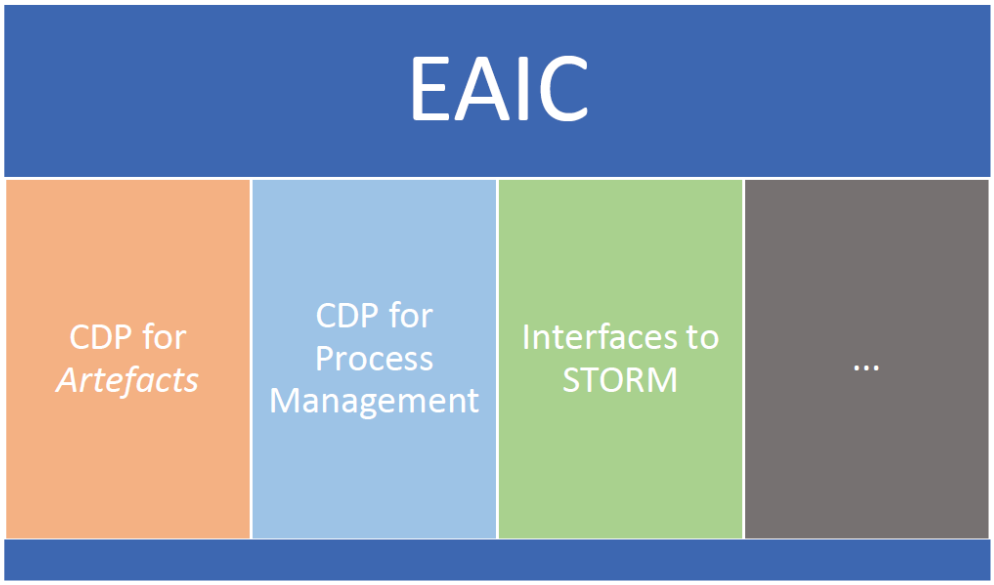


Figure 14. The CDP types that are parts of a EAIC (Source: authors)

EA views are selected parts of models, services and CDPs, representing a complete *Entity* and its ICS architectures; where the focus is on aspects that address the: 1) Tangible concerns of one or more stakeholders, by using STORM; and 2) Intangibles, which is mainly quality. The

EAIC supports the design of such complex *Entity* models to support a view, using ARCP, SARP, BPMP, RESP and many others. From an EA point of view, the MVC is one of the most used and known patterns.The EAIC can be used to abstract EA artefacts for *Project*s**,** where *Project* engineers implement interconnected patterns. The complexity, diversity and cross-functional nature of EA based MDTCAS and other *Project*'s domains require that various categories of patterns should be developed and classified in various disciplines, domains, and levels of precision. The integration of various categories of patterns and their non-standardization cause to this topic lack of maturity [30], that is why there are needs for a CDP concept in the MDTCAS.

Common Denominator Patterns

Many *Entities* are applying patterns to abstract their EAs or Project methodologies at various levels ranging from software design patterns, business patterns to enterprise patterns. There is no single standard for describing EAIC, so this article can be considered as a pattern for abstracting existing major BBs and pattern categories related to EAIC. The EAIC is a sum or set of CDPs, where CDPs may have OO-like relationships; and it is in fact a pattern for integration of in-house and standard patterns. The EAIC contains the following set of CDPs:

- The CDP for Services (CDP4S).
- The CDP for Intelligence (CDP4I).
- The CDP for Knowledge (CDP4K).
- The CDP for Interfaces (CDP4I).
- The CDP for Data (CDP4D).
- …..

The *Project* needs a specific integration process for EAIC, *Artefacts* and MDTCAS.

EAIC,Artefacts and MDTCAS-Integration Process

Generic Characteristics

Generic Building blocks (GBB) have the following generic characteristics [33]:

- It is a functionality defined package to meet *Project*'s requirements.
- It has published interfaces to access the defined functionalities.
- It may interoperate with other related *Artefacts*.
- The optimal *Artefact* has the following characteristics: 1) It facilitates implementation and maintenance to integrate ICS and related standards; 2) It may be assembled from other *Artefacts*, hence patterns; 3) It can be a subassembly of other *Artefacts*, hence patterns; and 4) An*Arteface*t is re-usable and replaceable in any environment.
- It may have multiple implementations, with probably different inter-dependent *Artefacts*.
- It is a package of functionality (a library) used for APD/business requirements.

One form of a GBB is the systemic *Artefact* that contains systemic characteristics, like error management, security, manageability, persistence… They are pervasive in all *Project*'s components [33]. A *Project* must define the manner to assemble patterns, functionalities, tools, and other *Artefacts*into CDPs; to avoid lock-in, by using PPMC. The PPMC is an *Entity* to define its EAIC and the way it implements its *Artefacts*, which improves the way how legacy systems are transformed into dynamic systems.

Dynamic Systems Transformation

Dynamic systems transformation, using a RP, PPMC and STORM to:

- Transform through therefinement: *Entity*'s ICS are a collection of *Artefacts*, which are result of a RP and the use of a standard set of BBs. *Artefacts*must interoperate with other *Artefacts*, and it is important that their interfaces are stable. *Artefacts* can be defined at various levels of maturity, depending on the *Project*'s evolution. In RP early phases, an*Artefact* can be an interface to functionalities which use legacy components. *Artefacts* basics are defined in EA methodologies like TOGAF, as ABBs. As the *Project* advances, complex implementations replace these basic definitions of functionality, to become SBBs [33].It is recommended that an *Entity* develops its own version of the SBB, which is its *Project*'s BB (PBB), which contains its PPMC characteristics and interfaces to STORM.
- ABBsrelate to the Architecture Continuum and are managed by the ADM. Its main characteristics are according to [33]: 1) Define the needed functionalities; 2) Capture Business and ICS requirements; 3) Make the *Project* technology aware; and 4) Manage SBBs' implement process. ABB's specifications include: 1) Fundamental functionalities and attributes; 2) Interfaces; 3) Mappings to *Entity*'s strategy policies to STORM; and 4) Related *Artefacts*, like SBBs, with detailed information.
- SBBs relate to the Solutions Continuum and may be either external or internal; and their main characteristics are according to [33]: 1) Define which patterns will implement which set of functionalities; 2) Define pattern's implementation details; 3) Fulfil *Project*'s business requirements; and 4) Be product- or vendor-aware, by applying the PPMC and STORM. *Artefact*'s specifications include: 1) Specific functionalities and attributes; 2) Interfaces; 3) Required SBBs; 3) Mapping of the used SBBs to ICS' topology and operational policies; 4) Specifications of attributes shared across the ICS; 4) Performance tuning; 5) Design drivers and constraints, including the physical architecture; and 6) Relationships between SBBs and ABBs, and the evolution of the SBB towards the PBB.
- *Project*'s Building Blocksextend the *Project*'s EA concept, to become a PBB. A PBB supports the categorization of patterns to implement the needed transformed components. PBBs are a combination of software and platform patterns, like connectors which serve as a glue that connects various types of components. A PBB is a set of related BBs, used to put together a component and support a business service or a service[33].
- Atomic Building Blocks Concept: Today's dynamic *Entities* have to struggle for survival, and they must be loosely interconnected in a global market. It is not a secret that a solid business environment that wants to ensure its sustainable business future must adapt itself to frequent *Project*s, to adapt to such a situation, service-based solution is proposed to support the *Project*'s main artefacts like the ABB. Such a service-based strategy for frequent changes is translated into a set of solutions in the form of SBBs, supporting the continuous improvement of various business and ICS resources. Agile and loosely coupled ABBs can be used to improve the quality and success rate of the implementation and integration of the defined *Project*'s requirements. That is achieved by simplifying and unifying of the used sets of applied *Artefacts*under EAIC's umbrella, which can be used for the Analysis, Design, Development, Tests, and Maintenance sub-phases. The optimal EAIC is based on the 1:1 mapping in which each requirement and its artefacts, like services/sBBs, are totally independent. Standardized and simplified enterprise business architecture, enables the *Project* to become iterative, where its design is based on the ADM. The services resources traverse through the ADM, where each phase refines the service's implementation's capability; such an approach uses a holistic view on the ICS that consists of: 1) A unified collection of services, used to implement needed

components, 2) services-based data and software components, and 3) Scalable technology infrastructure. The coordination of these main ICS resource parts is insured by the use of: 1) EAIC; 2) TOGAF/ADM and UML, and 3) Efficient tools. Managers use this concept to gain knowledge on how a *Project* can be managed, using services, and sBBs. Such a *Project* has to make a choice of the optimal tooling and modelling environment based on a pseudo-Model View Control pattern. The complexity of the *Project*'s implementation phase often causes the *Project*'s failure, and failure rates are very high. The EAIC supports a cross-functional transformation process based on: 1) Requirements engineering and interfaces to *Factors*; 2) Business Architecture that includes STORM; 3) BPMs; 4) SOA; 5) *Entity*'s organizational structure (or organizational engineering); 6) the ICS' structure; and 7) Continuum's integration[34,35].

- atomic Building Blocks Structure, the services comply with TOGAF's generic characteristics of BBs which have the following characteristics [36]: 1) It is a package of requirements, functionalities and artefacts designed to meet *Project*'srequirements; 2) It has standardized interfaces; 3) It is interoperable with other types of BBs and can be an aggregation of other services; 4) It defines the functionalities that will be implemented and *Project*'s requirements; 5) It ensures technology awareness and respect of standards; 6) It can be used as a template to implement/instantiate sBBs; 7) It is a reusable and replaceable template; that serves CDPs; 8) It can have many implementations; and it has a GUID and respects the 1:1 mapping concept.

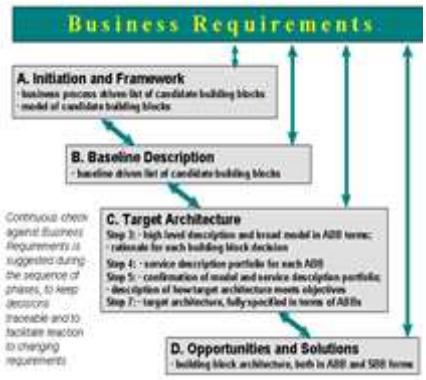


Figure 15. BBs' management in ADM's phases [36].

An *Artefact*is an architecture element, or a package of functionalities and resources designed to meet *Project*'s transactions. The way artefacts, functionalities and development resources are combined in an *Artefact* might vary. The *Project*'s team must coordinate the design and prototype of services, using the ADM's various phases, as shown in Figure 15; where these *Artefacts*/services will transform the legacy components, facilitate integration and enable *Artefacts*/services' interoperability [36]. *Artefacts*support the *Project*'s unbundling of its monolithic environment by breaking the previous legacy components into a set of classified unique sets of *Artefacts*. *An Artefact is just another building brick in the Entity's wall...* The *Project*'s team builds a PoC to define the needed sets of services during the unbundling process. But the system's dynamic transformation depends on the evolution of *Artefacts/services* architectures.

The Evolution of *Artefacts*/Services' Architectures

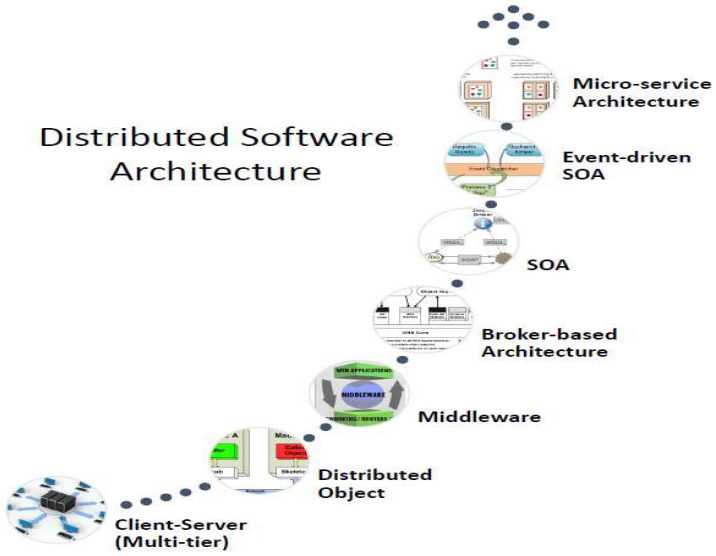


Figure 16. The evolution of distributed ICS and Artefacts/services architectureToday, we have mainly service SOA and MSA based Artefacts, but as shown in Figure 16, it is the result of ICS based modular actions development. In SOA an Artefact/service can be composed of other Artefacts; whereas in MSA a service is independent and is self-contained and that implies that it cannot be composed of other services. This is one of the main differences between SOA and MSA based EAIC. In fact, SOA and MSA are extremely similar, except for BPM's degree of encapsulation, where a BPM contains sets if services needed to complete the task. In an MSA this would be a conflict in purpose. This implies that MSA is really a subset or special architectural form of SOA. MSA provides an approach to delivering SOA in an effective manner for the right set of business drivers, that can be evaluated by STORM [53].

Enterprise Integration Pattern's CSFs

Based on the AHMM4PPMC, LRP4PPMC and DMS4PPMC, this CSA's CSFs/KPI were weighted by the HDT.eval function and the results are shown in Table 16. This CSA's result of 8.20, which is low and insufficient.

Critical Success Factors	AHMM4CBB: KPIs	Weightings
CSF_Enterprise_Patterns_EA_Integration	Feasible	From 1 to 10. **09 Selected**
CSF_Enterprise_Patterns_Extracting_Artefacts	Complex	From 1 to 10. **08 Selected**
CSF_Enterprise_Patterns_API/REST/ESB_Patterns	Complex	From 1 to 10. **08 Selected**
CSF_Enterprise_Patterns_EAIC_CDP	Complex	From 1 to 10. **08 Selected**
CSF_Enterprise_Patterns_MDTCAS_Integration	Complex	From 1 to 10. **08 Selected**

valuation

Table 3. This CSA has the average of 8.20.

This is mainly becausethe Enterprise Integration Pattern is complex. As this CSA presented positive results, the next CSA to be analysed is EA's approach for STORM.

THEENTERPRISE ARCHITECTURE APPROACH FOR STORM

Using a minimalEA (for the *target architecture*), can support*Project*s to align theirplans with architecture visions and STORM. The traditional architecture layers represent a silo concept, where it is complicated to transform them into an agile transformed system. Using thePPMC, the*Project* transforms the legacy ICS into an agile classified directory of *Artefacts*/services

[8]. The*Manager* and the *Project* team must have in depth ILP/knowledge of various EA, ICS, and DMSbased STORM domains, which completes the profile of an Architect of Adaptable Information System (AofABIS), who uses the PPMC and STORM. The first step is to assess with STORM the readiness for *Project's* readiness.

Assess with STORM Project's Readiness

A *Business Transformation Readiness Assessment* evaluates and quantifies the *Entity's* readiness to change and start a*Project*, by using STORM. The STORM based*Project's*assessment is based on PPMC readiness *Factors*. The outcomes of theSTORMbased readiness assessment is added to the *Capability Assessment*. These outcomes use STORM to establish the ADM and DMS's interfaces, to support the*Project*, and to localize risk constraints. These *Factors* associated with the *Architecture Vision* are related to the initial level of risks, like: catastrophic, critical, marginal, or negligible. These *Factors*for STORM are to integrate and coordinate the ADM.

ADM's Coordination

PPMC's integration in projects is done by using the ADM, which supports it in the automation of DevOps activities, especially to manage *Artefacts*. Throughout ADM phases,PPMC phases are created or improved. The ADM encloses cyclic iterations, where all PPMC's instances actions are logged. PPMC is domain agnostic and technology (monolithic or services) independent. PPMC's integration with the ADM has the following advantages, to achieve: 1) Real-time transformation, mapping, and Microartefact/services management; 2) Improving ofICS' performance, and robustness; 3) The PPMC enabling the use of standard methodologies like UML or ArchiMate; and 4) Tests and integration-driven developments approach.

PPMC Enabled Tests

PPMC's must check if requirements respect [57]: 1) Completeness, where they must contain all needed information and *Artefacts*; 2) Clearness, where they should be transparent and clear; 3) Correctness, where all contents must be credible, evaluated by STORM; 4) Consistency, where they should not contradict other requirements; 5) Testability, validates the PIP requirements' sets. The ADM controls, directs, and monitors PIP, mapping and *Artefacts*by using PPMC adapted set of tests and integration-driven developments. These tests are:

- TDD for PPMC (TDD4PPMC): The standard for unit tests (or TDD) is asemi-manual concept used in PIP development (known as the *test first approach*), where a TDD can be attached to a class that represents a service, or a set of legacy code [37]. Design Driven Development (DDD) is mainly used for MSA (or the model first approach),which is based on designing first the model/solution that containsproject's requirements, mappings and Microartefacts.Other model first methodologies are UML, ArchiMate's (or other) UCs where each UC maps to a concrete set of diagrams. The class diagram maps to requirements and Microartefacts/services where the UC defines unit and integration tests. Automated tests evaluate PPMC models for a given set of requirements and verifies their statuses. The DDD or other model first methodologies need Acceptance Test Driven Development (ATDD) mechanisms [38].
- ATDD is applied in the case of collaborate business clients, project testers and EA/PIP engineers, to assist their communication [39]. Based on standard TDD s, the ATDD is based on developing tests, which represent the results of the requirement's behaviour and their corresponding sets of Microartefacts/services. Business users contribute to model credible acceptance tests or use BDD techniques [40].

- BDD: The PPMC can usethe BDD that includes unit, integration and acceptance tests [41]. The BDD has a pseudo-prose formalism that resembles to logical human scripts, so that business specialists implement UC scenarios and their corresponding tests. The BDD includes a resources' mapping subsystem to link pseudo-prose keywords to Microartefacts/services, and to business requirements. PPMC enables an automated SBBs/services' testing environment that is used with ADM's iterations [42]. There are many modelling languages, but probably for EA, ArchiMate is the most advanced one.

Modelling Language-ArchiMate

The *elicitation* of *Project's* requirements and their corresponding and mapped services is the first activity or step in the transformation process in the context of ADM. *Elicitation* refers to the capture of requirements, to avoid the assumption that *Project'*srequirements are ready to be simply collected, by using basic technics. Information gathered from the transformation process *elicitation* has to be interpreted, analysed, modelled and validated before EA specialists, who can confirm that a credible and coherent set of project*'s* serviceshas been located and mapped. Therefore, *Project'*srequirements and services elicitation is directly related to all ADM activities. The *elicitation* discipline used is dependent on the used modelling scheme, and vice versa. Modelling schemes can imply the application of specific elicitation techniques, like the ones used with ArchiMate [43]. EA based modelling using ArchiMate has the following characteristics [44]: 1) Models behavioural and structural elements of a *Project*; 2) Enables EA modelling to support ICS infrastructure and landscapes; 3) EA models implemented using ArchiMate can be stored in the project*'s* repository; 4) Model data/information behaviours; 5) Use an interchange format based on the Mark-up Language (XML) which can map to ArchiMate's Model Exchange File Format's XML schema(s); 6) Schema's properties are mapped to instances of ArchiMate property definitions; and 7) The used models should generate a map to ICS' and its applications' cartography.

Application Cartography

SBBs and services map to the*Entity's* cartography of applications, where the*Entity's* applications are classified as follows [45]: 1) Using EA capacities, like TOGAF's Application Communication Diagram (ACD), which depicts its used models and mappings related to communications between applications and modules (of services), in form an *Entity's* metamodel. It presents applications, components, and interfaces (between various components and services); 2) Interfaces may be associated with data classes, applications can be related to Microartefacts/services; 3) Application communication diagrams can represent, an existing applications' cartography, or a logical architecture of the transformed end-system.

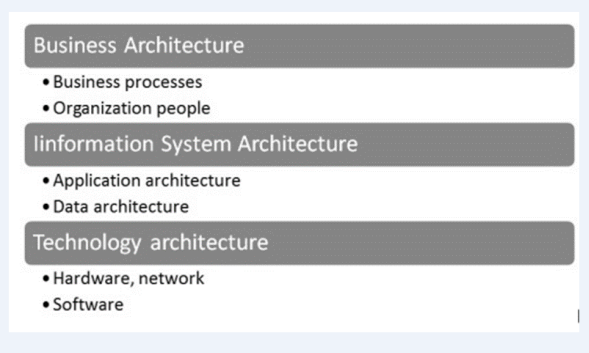


Figure 17. The architecture is layered [45]

Artefacts based EA is privileged; 4) *Entities* have hybrid (mixed) applications, repositories and new *Artefacts*based EA; 5) In the case of using *Artefacts*, services based application components, should be structured according to their nature and their EA level; 6)

Microartefacts based components are related to services, which use connectors; 7) A dimension of the applications' cartography should be dedicated to EA model's integration to support STORM; and 8) As shown in Figure 17, the EA is layered, where the interaction component layer is on top, process based components in the middle, and entity components on the bottom.

Architecture Layers

PPMC's architecture helps in establishing EA principles that are defined in the project's preliminary phase and it guides its vision. The project*'s* EA superposes existing architecture standards, like TOGAF, as shown in Figure 17[8]; the PPMC is a tailored adoption of TOGAF and defines this approach as a just-enough EA, as shown in Figure 4, with the following PPMC layers are: 1) Business Architecture; 2) Data Architecture; 3) Application Architecture; and 4) Technology Architecture.ABBs and SBBs are used to solve assembled Microartefacts and to support EA principles, these blocks are a set of project's deliverables. The dimensions of EA are scoped to project*'s* boundaries, which have to consider the heterogeneous types of services' architectures and legacy systems [8]; which could be supported by: 1) Defining deliverables/templates; 2) Defining EA's interactions; 3) Applying*Artefacts*integration; 5) Applying a modelling strategy approach; and 6) EA and STORM based Transformation Risk Management.

EA for STORM's CSFs

Critical Success Factors	HMM enhances: KPIs	Weightings
CSF_EA_STORM_Assess_Readiness_Tests	Proven	From 1 to 10. **10 Selected**
CSF_EA_STORM_ADM_Modelling	Possible	From 1 to 10. **09 Selected**
CSF_EA_STORM_Cartography	Possible	From 1 to 10. **09 Selected**
CSF_EA_STORM_Layers	Possible	From 1 to 10. **09 Selected**
CSF_EA_STORM_DMS_Interfacing	Complex	From 1 to 10. **08 Selected**

valuation

Table 4. This CSA has the average of 9.0Based on the AHMM4PPMC, LRP4PPMC and DMS4PPMC, this CSA's CSFs/KPI were weighted by the HDT.eval function and the results are shown in Table 4. This CSA's result of 8.20, which is high and sufficient. This is mainly due to the fact that the EA'sfor STORM is implementable. As this CSA presented positive results, the next CSA to be analysed is the STORM based DMS.

THE STORMBASEDDMS

ActualProject relatedPPMC, DMS, EA/ADM, PIP, DevOps (development, operations), integration, security, and test IDEs are skeletons that use heterogenous scripting environments, subsystems, and methodologies, which do not offer a unified transformation strategy. That is why there is a need for STORM, which can use the DMS, which in turn depends on HDT scenarios.

HDT Scenarios

Intelligent scenarios, which can be implemented using: 1) An interaction of Artefacts/services; 2) BPM instances; 3) HDT and Factors; 4) IHI framework; and 5) Other…HDT scenarios depend on the unbundling of the legacyenvironment and its monolithic ICS, which offers sets of automatized *Artefacts*which can be used in these scenarios, like BPM(s)[46]. The unbundling process upstreams intelligent scenarios which are not altered to integrate traditional services and aligns with the ADM. At theProject's start, ICS and EA specialists create the top-level organizational design *Artefacts* which are used to create the classification concept and that becomes a point of reference for Intelligent APD/business scenarios' andprocesses. This classification concept is used to classify the

requirements, *Artefacts*, and BPMs. Unbundling and transforming capabilities, refer to the notions of atomic, unique, meaningful granular unbundling processes that generate intelligent business *Artefacts*to serve dynamic/intelligent scenarios.This enables the discovery of functional capabilities, and to avoid duplicating APD/business capabilities. From DMS' perspective, *Artefacts*are the result of transforming the ICSlegacy system, so that it enhances its decision-making and STORM capabilities. Agility is crucial for intelligent dynamic systems, where *Artefacts*unbundling strategy's aim is transforming monolithic functions into portfolios of granular Artefacts, by using abstract Artefacts' interfaces concept. The RP based unbundling of APD/business activities and their decomposition in the form of intelligent HDT Scenarios that can be used, filtered, traced and queried; are stored in the *Entity'*scatalogues. Catalogues contain the following entities: 1) Organizational units' information and business function; 2) Intelligent business Scenarios, their information service equivalence, and APD activities and STORM ILPs[36].

APD Activities

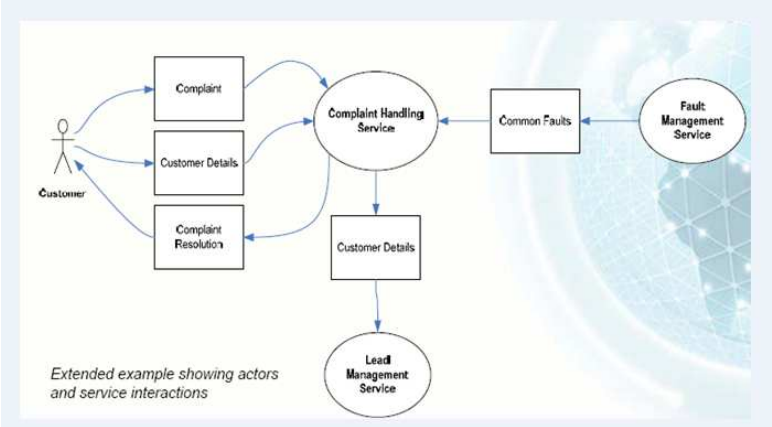


Figure 18. Business modelling and business services' Interaction [47]

STORMimproves the *Entity*'s sustainability and would improve problems management. EA based DMS' modelling strategy delivers *Artefacts* to support modelling languages; andthat offers a generic approach, what makes the *Project* independent and recommends the following APD activities: 1) APD/Business cases or UCs modelling, is the starting point for any new requirement,*Artefacts* usage, and STORM setup; 2) The mapping concept supports the UCs; 3) A UC maps to a BPM, which links to a set of *Artefacts*, as shown in Figure 18, and STORM structures: STORM2CSA, CSA2CSF, CSF2KPI, and KPI2VAR [47]; 4) STORM the business architecture, focusing on intelligent scenarios and *Project's* strategy; 5) EAand STORM support the design of new APD/business activities and intelligent data/ILP/knowledge *Artefacts*.

Intelligent Data, ILP, and Knowledge Artefacts

STORM contains ILP data-models, related modelling components and does not depend on the types of data-sources; but their diversity generates problems, especially in the PIP. APD/business data *Artefacts*focus primarily on the encapsulation of the data schema(s) [48]. The mapping concept is applied for business data models management and access, where the sets of requirements correspond to a data *Entity* or a *business data view*, if the data can be encapsulated in a single class; which facilitates the usage of intelligent scenarios like BPM. A BPM Oriented Knowledge (BPMOK) management framework can be applied for knowledge management, which supports the DMS and STORM. STORM should offer and control the optimal DMS Architectural Model (DMSAM).

The Optimal DMSAM

A simplified and unified PPMC must be used for the *Project*, because there are many standards and methodologies, which are used in parallel. That can cause parallel and siloed

solutions and ICSs; where the PPMC can relate various fields like Cloud architecture, EA, SOA, JEE, UML… *Entities* need the possibility to customize *Project's* modelling process and include standards, notation and information relevant to its structure. This concept results in the extension of modelling capabilities of the environments like ArchiMate, UML by the usage of their extension mechanisms. The optimal approach is to use common *Artefacts* and CDPs [49] … A useful modelling environment can be System ML (SysML), which is the result of UML's evolution and has brought significant capabilities. The PPMC can be used for the DMSAM, and that supports: 1) DMS and STORM's requests; 2) DMS4PPMC and STORM problem and solution sets; 3) Properties and interfaces; 4) Levels of granularity; 5) Implementing traceability (derived and source); 6) Constraints and verification test cases; and 7) Factors manipulation and weighting.

STORM based DMS4PPMC' CSFs

Critical Success Factors	KPIs	Weightings
CSF_STORM_DMS_HDT Scenarios	Complex	From 1 to 10. **08 Selected**
CSF_STORM_DMS_APD_Activities	Complex	From 1 to 10. **08 Selected**
CSF_STORM_DMS_Data_ILP_Knowledge	Complex	From 1 to 10. **08 Selected**
CSF_STORM_DMS_DMSAM	Complex	From 1 to 10. **08 Selected**

valuation

Table 5. This CSA has the average of 8.0.

Based on the AHMM4PPMC, LRP4PPMC and DMS4PPMC, this CSA's CSFs/KPI were weighted by the HDT.eval function and the results are shown in Table 5. This CSA's result of 8.0 is low and insufficient. This is mainly due to the fact many complex interactions. As this CSA does not present positive result, and the next is to execute the PoC.

THE PROOF OF CONCEPT

The Structure

STORM's PoC was implemented using the research's framework that had been developed using the framework*'s* Natural Language Programming (NLP), Microsoft Visual Studio .NET, C/C++ and Java. The PoC is based on the DMS4PPMC/AHMM4PPMC and the CSFs' binding, using specific *Factors*, where the STORM was designed using an UML and TOGAF methodologies. The PoC has the following structure and execution steps, as shown in Figure 4:

- Building an IHI transformation framework.
- Building an IHI PPMC.
- Preparations include linking a scenario to *Artefacts*, as shown in Figure 19.
- Execute Phase 1-a, which contains the literature review, Tables (CSAs) evaluations, and delivers the decision to continue to (or not) Phase 2. After selecting the CSAs/CSFs tags are linked to various STORM *Artefacts* scenarios. Links Factors to structures, like CSA2CSF…
- Execute Phase 1-b, correlates Tables 1-to-5 and presents a synthesis.This concludes Phase 1.
- Execute Phase 2-a, which the DMS4PPMC and its HDT to deliver possible STROM solutions. The HDTused an AHMMPPMC instance and intelligent scenarios,
- Execute Phase 2-b, solve a concrete problem.

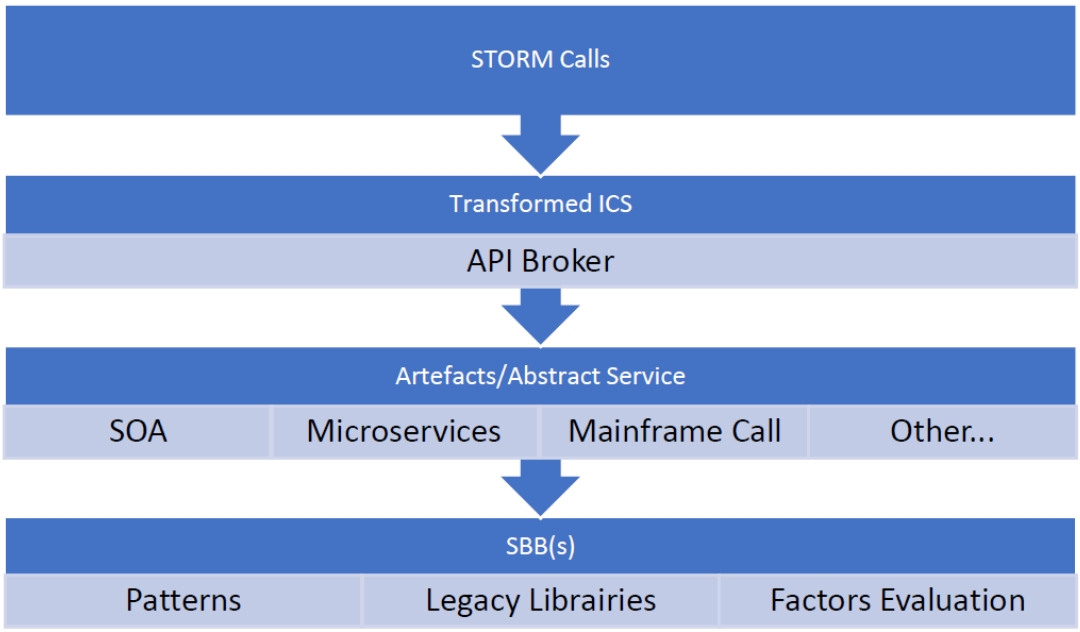


Figure 19. Artefacts integration for STORM.

The research maps *Factors* to sets of *Artefacts*, as shown in Figure 20, and then to the following STORM structures: STORM2CSA, CSA2CSF, CSF2KPI, and KPI2VAR. The STORM uses *Factors*, GUIDs and the DMS4PPMC to support the *Project's* strategy. Then the IHI framework's frontend mapping/linking actions are activated by: 1) Selecting an HDT node that contains the *Factors*, and 2) Selecting the problem to be solved using NLP.

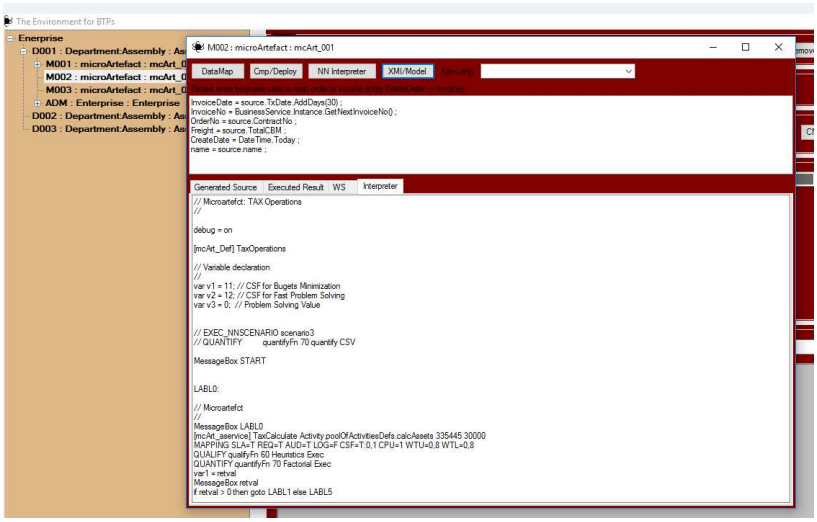


Figure 20. The NLP interface.

The STORM uses DMS4PPMCandILP/knowledge database to generateactions to solve a concrete strategy requests orproblems. Once the IHI framework' development interface is activated, the NLP interface can be launched to implement STORM scripts, as shown in Figure 20. These NLP based STORM scripts that make up the KMS4PPMC/DMS4PPMC subsystem relate to a set of RP generated*Artefacts*. STORM-related *Factors* were selected as demonstrated previously in this article's tables and the result of the processing of the DMS4PPMC, as illustrated in Table 6, shows that STORM is not an independent topic and is strongly bonded to the *Project's* overall strategy related risk management concept.

CSA Category of CSFs/KPIs	Transformation Capability	Average Result	Table
PPMC'S Appliance	Usable-Mature	From 1 to 10. 9.25	1
ICS and services	Transformable-Possible-Complex	From 1 to 10. 8.30	2
Enterprise patterns integration	Transformable-Possible-Complex	From 1 to 10. 8.20	3
EA for STORM	Transformable-Possible-Mature	From 1 to 10. 9.0	4
STORM based DMS	Heterogenous-VeryComplex	From 1 to 10. 8.00	5
Evaluate First Phase			

Table 6. The STORM research's outcome.

The model's main constraint is that CSAs having an average result below 8.5 will be ignored. As shown in Table 6 (which has a rounded average of 8.50), this fact keeps the CSAs (marked in green) that helps make this article's conclusion; and no CSA in red colour. It means that STORM integration will succeed and that the Project must be done in multiple transformation sub-*Projects*, using the ADM4PPMC, where the first one should try to transform the base systems, the ICS.

The APD Case for STORM and Setting Up SWOT

The APD case is built around ArchiSurance [50], where the central point is the feasibility of the STORM based strategy (as shown in Figure 20)and the usage of the PPMC.

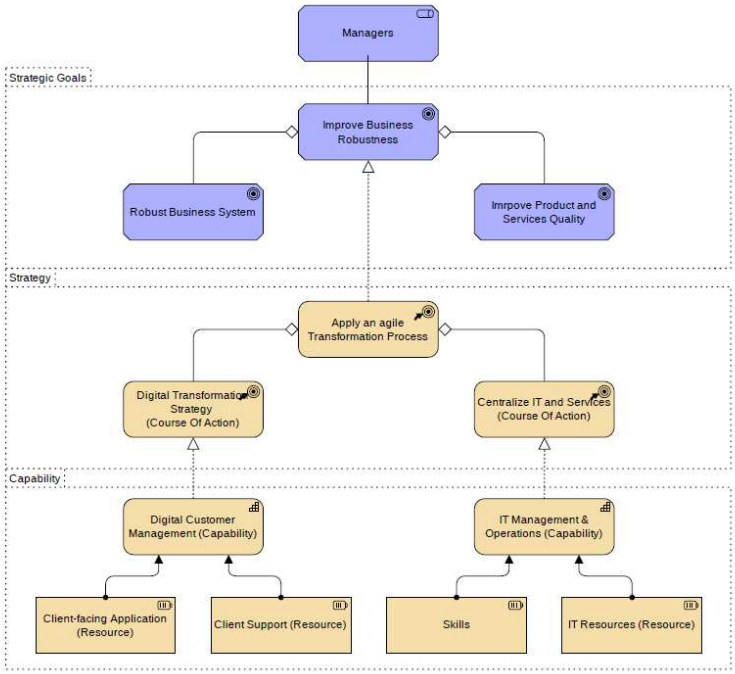


Figure 20. The STORM based strategy view

The SWOT analysis is one of ArchiMate's strategy views and it supports the evaluation of *Project's* internal strengths and weaknesses, which includes needed resources and current capabilities, which can enable (or hinder) *Project's* PIP and default its strategies. The PPMC can present external opportunities and threats which enable (or hinder) *Project's* performance. The SWOT analysis view supports the design of strategies that are aimed to exploit deduced

Os and Ss, while proactively predicting possible Ts and to eventually improve Ws, and applying the following strategies [51]:

- WT strategy: minimize both Ws and Ts, to enable the achievement of defined PPMC.
- WO strategy: minimize Ws in order to be able to take advantage of new Os.
- ST strategy: maximize Ss in order to be able to deal with Ts.
- SO strategy: maximize Ss, using existing resources to promote new Os.

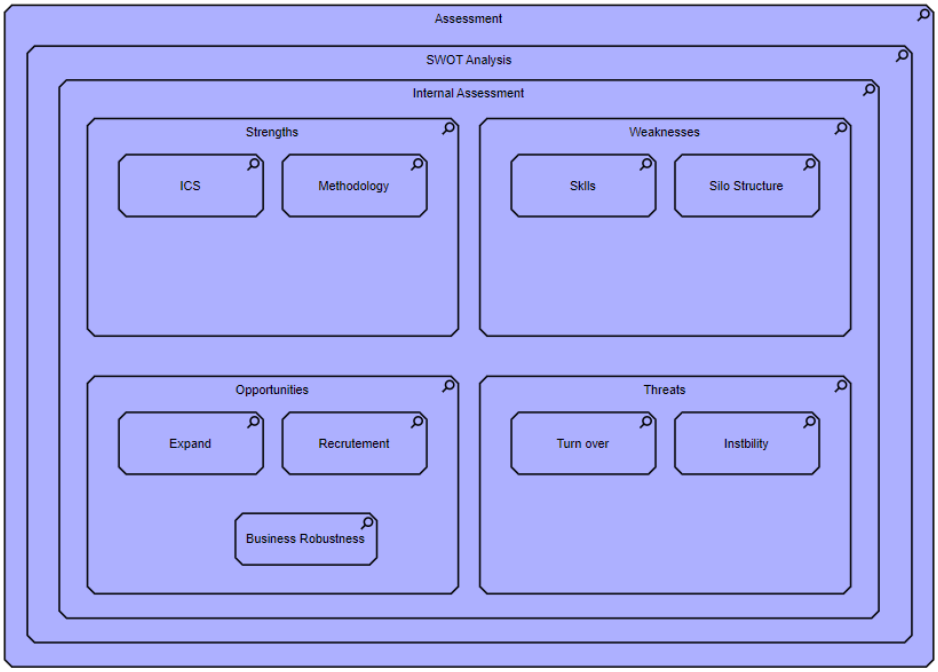


Figure 21. SWOT's strategic view

After setting up STROM, the next step is to related SWOT analysis to intelligent scenarios and corresponding Artefacts.

Setting Up STORM Intelligent Scenarios

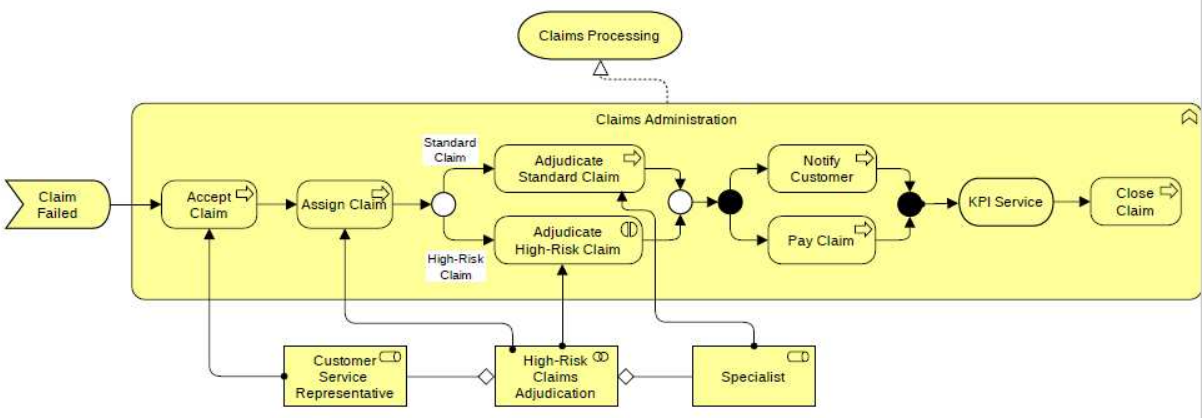


Figure 22. An intelligent scenario

An intelligent scenario, as shown in Figure 22, is a BPM that is interfaced with *Artefacts*, which in turn link to structures: STORM2CSA, CSA2CSF, CSF2KPI, and KPI2VAR. The STORM uses intelligent scenariosto evaluate a strategy and deliver SWOT values. It can be also used for solving concrete requests or problems.

Solving a Concrete Request or Problem

In Phase 2, the HDT is used, to find a combination of heuristics action, used to solve a problem related to the RQ. A selected CSF is linked to a problem type and a related set of actions where the processing starts in the root node. Each problem, like this case the PRB_STORM_Strategy_Deviationproblem, has the following set of actions:

- ACT_STORM_Strategy_Deviation_ Define Possible Corrective Processing
- …

For this DMS4 PPMC related PoC, the authorshave selected the CSF_STORM_Strategy_Deviation_ Validation as the active CSF, taken from the CSFs pool. In this PoC the goal is to find solutions related to this selected CSF's related problems. The authors have decided to apply the AHMM4PPMC based reasoning to try to solve the CSF_STORM_Strategy_Deviation_ Validation issues and the related problem or the PRB_STORM_Strategy_Deviation_ Validation, which is solved by using the following steps:

- Relating the APD case and integration capabilities to CSF_STORM_Strategy_Deviation_ Validation capabilities that was done in Phase 1.
- Link the processing of this node to the pseudo-quantitative modules, then by using qualitative modules, filter and deliver the initial state that is the root node of the decision tree.
- The HDT is configured, weighted and tuned using configuration information.
- The set of possible solutions results from the DMS4PPMC. Then the HDT is launched to find the set of possible solutions in the form of possible strategy improvements.
- The NLP scripts make up the processing of STORM's logic and is supported by a set of predefined actions. These actions are processed in the IHI framework background to support *Artefacts* that are called by the HDT.

CONCLUSIONS AND RECOMMENDATIONS

The set of STORM's architecture, refinement, technical and managerial recommendations:

- Implement an IHI framework.
- SWOT risk analysis is a basic and a highly technical methodology that can be used to check *Project*'s strategy, capabilities or BU's viability.
- Link STORM to SWOT and *Factors.*
- This chapter presents the possibility to implement an IHI STORM which avoids the financial-only locked-in strategies and ensures success.
- RP is a *Project's* critical phase, and a*Project* must build a holistic MDTCAS to support the RPs activities.
- Building a flexible and scalable ICS.
- The PPMC needs to define a MDTCAS manages RP's basic elements: *Artefacts*. The major innovation in this article is linking of the *Managers* popular risk and quality management (like SWOT, Six Sigma, …) to concrete components.
- Each *Entity* constructs its own IHI STORM.
- The STORM replaces legacy-solutions using conversion concepts in order to ensure *Project's* success.
- STORM interface *Entity's* TDM and delivers the pool of *Artefacts*.
- The ADM manages design, RP, DevOps, and PPMC activities.
- *Entity's Artefacts'* stability and coherence are crucial for its evolution.
- Avoid consulting firms and to build internal STORM.
- STORM's integrationis very complex and will very probably face failure.

- PPMC and DMS4PPMC, based on pseudo-SWOT approach, which uses *Factors*.
- Gained knowledge/experience can be fed in the *Entity*'s KMS; and that is how it builds its own ILP.
- The PPMC is a *Model First Approach* that uses a pseudo bottom-up approach.
- The PPMC provides a concept for classifying and using existing types of *Artefacts*.
- Using a minimal EA (for the *target architecture*), can support *Project*s to align their plans with architecture visions and STORM.

REFERENCES

[1] Remawati, D. "Analisis SWOT Implementasi Green Computing Di SekolahKejuruan (StudiKasus Pada SMK XYZ)," J. Ilm. SINUS, no. ISSN:1693-1173, pp. 23–36, 2016.
[2] Wati, A., Ranggadara, I., Kurnianda, N., Irmawan, D., &Frizki, D. (2019). International Journal of Innovative Technology and Exploring Engineering (IJITEE). ISSN: 2278-3075, Volume-8 Issue-12, October, 2019. Blue Eyes Intelligence Engineering & Sciences Publication. DOI: 10.35940/ijitee.L2772.1081219.
[3] Kitsios, F., Kyriakopoulou, M., &Kamariotou, M. (2022). Exploring Business Strategy Modelling with ArchiMate: A Case Study Approach. MDPI. doi.org/10.3390/info13010031 https://www.mdpi.com/journal/information
[4] Ylimäki T. (2006). Potential critical success factors for EA. Journal of Enterprise Architecture, Vol. 2, No. 4, pp. 29-40.
[5] Trad, A. (2023). Organizational and Digital Transformation Projects-A Mathematical Model for Building Blocks based Organizational Unbundling Process. IGI Global. USA.
[6] Liu, A. (2022). Rumbaugh, Booch and Jacobson Methodologies. Opengenus. https://iq.opengenus.org/rumbaugh-booch-and-jacobson-methodologies/
[7] Rosing, M., Hove, M., Subbarao, R., & Preston, T. (2012). Combining BPM and EA in complex ERP projects (a Business Architecture discipline).
[8] The Open Group (2011). Introduction to the Architecture Development Method (ADM). The Open Group. USA.
[9] Patni, S. (2017). Pro RESTful APIs Design, Build and Integrate with REST, JSON, XML and JAX-RS. Apress. Santa Clara, California. USA.
[10] Trad, A. (2022). Enterprise Transformation Projects-Cloud Transformation Concept – The Compute System (CTC-CS). IGI Global. USA.
[11] Sullivan, D. (2020). Official Google Professional Cloud Architect-Study Guide. John Wiley & Sons, Inc. USA.
[12] Wikipedia (2022a). Cloud computing. The Wikipedia. https://en.wikipedia.org/wiki/Cloud_computing
[13] Cloud Computing Patterns (2022). Private Cloud. https://www.cloudcomputingpatterns.org/private_cloud/
[14] Oracle (2016a). API Management in 2026. Oracle. USA.
[15] Oracle (2016b). Evolution and Generations of API Management. Oracle. USA.
[16] Sidell, E. (2020). Choosing the Right API Gateway Pattern for Effective API Delivery. NGINX. USA. https://www.nginx.com/blog/choosing-the-right-api-gateway-pattern/ , Accessed on January 26th, 2022
[17] Fowler, M. (2003). Catalog of Patterns of Enterprise Application Architecture. http://martinfowler.com/eaaCatalog, Accessed on January 26th, 2022
[18] Fowler, M. (2014). Microservices. http://martinfowler.com/articles/microservices.html. USA.
[19] GitBook (2021). ArchiMate Guide-ArchiMate and TOGAF Layers. GitBook. https://archimatetool.gitbook.io/project/archimate-and-togaf-layers, Accessed on January 26th, 2022
[20] Farhoomand, A. (2004). Managing (e)business transformation. Palgrave Macmillan. UK.
[21] Walker, J. (2019). Generics & Design Patterns. Oberlin College Computer Science. https://www.cs.oberlin.edu/~jwalker/langDesign/GDesignPat/, Accessed on January 26th, 2022
[22] Buschmann, F., Meunier, R., Rohnert, H., Sommerlad, P., &Stal, M. (1996). Pattern-Oriented Software Architecture: A System of Patterns. Wiley. USA.
[23] Taleb, M., &Cherkaoui, O., (2012). "Pattern-Oriented Approach for Enterprise Architecture: TOGAF Framework". Journal of Software Engineering & Applications;Jan2012, Vol. 5 Issue 1, p45. Academic Journal .
[24] Crosswell, A. (2014). Bricks and the TOGAF TRM. National Institute of Health.
[25] Dong, J., & Yang, Sh. (2003). Visualizing Design Patterns With A UML Profile. Department of Computer Science. University of Texas. USA.
[26] Mapelsden, D., Hosking, J., & Grundy, J. (2002). Design Pattern Modelling and Instantiation using DPML. Department of Computer Science, University of Auckland, New Zealand.

[27] Burbeck, S. Application Programming in Smalltalk-80: How to use Model-View-Controller (MVC). University of Illinois in Urbana-Champaign (UIUC) Smalltalk Archive.
[28] Fuego (2006). BPM Process Patterns-Repeatable Designs for BPM Process Models Fuego
[29] Netapsys (2016). Les ESB, exemple particulier de Mule ESB. http://blog.netapsys.fr/les-esb-exemple-particulier-de-mule-esb , Accessed on January 26th, 2022.
[30] The Open Group (2018). Architecture Patterns. The Open Group. https://pubs.opengroup.org/architecture/togaf9-doc/arch/chap22.html, Accessed on January 26th, 2022.
[31] Bocanett, W. (2022). Break the monolith: Chunking strategy and the Strangler pattern-Build decoupled microservices to strangle your monolithic application. IBM. IBM, Tokyo Garage, IBM Tokyo R&D Lab. https://www.ibm.com/garage/method/practices/code/chunking-strategy-strangler-pattern/
[32] The Open Group (2018). Building Blocks. The Open Group. USA. https://pubs.opengroup.org/architecture/togaf9-doc/arch/chap33.html
[33] Radhakrishnan, R., & Rakesh Radhakrishnan, R. (2004). IT Infrastructure Architecture-Building Blocks. Sun Professional Services. USA.
[34] Trad, A. (2015). A Transformation Framework Proposal for Managers in Business Innovation and Business Transformation Projects-Intelligent aBB architecture. Centeris. Portugal.
[35] Trad, A. (2015). A Transformation Framework Proposal for Managers in Business Innovation and Business Transformation Projects-An ICS's atomic architecture vision. Centeris. Portugal.
[36] The Open Group (2011). Using TOGAF to Define & Govern SOA. The Open Group. USA.
[37] Janzen, D., &Saiedian, H. (2005). Test-driven development concepts, taxonomy, and future direction. Published in: Computer (Volume: 38, Issue: 9, Sept. 2005). IEEE.
[38] Design Patterns (2015). Design Patterns for TDD and DDD. The link was reviewed and extracted in March 2018, https://8408bcbcd6613c300fa58123cb291b2defe56766.googledrive.com/host/0Bwf9odcK3Cu0bFAzS3kzTDI4Tms/
[39] Koudelia, N. (2011). Acceptance test-driven development-Master Thesis. Uuniversity of Jyväskylä. Department of mathematical information technology. Jyväskylä. Finland.
[40] Koskela, L. (2007). Test driven: practical tdd and acceptance tdd for java developers. Manning Publications Co. Greenwich, CT, USA. ISBN: 9781932394856.
[41] Bingham, Ch., Eisenhardt, K., & Furr, N. (2007). What makes a process a capability? Heuristics, strategy, and effective capture of opportunities. Volume1, Issue1□2, Pages 27-47. Strategic Entrepreneurship Journal. Wiley Online Library. UK.
[42] Lazar, I., Motogna S., &Parv, B. (2010). Behaviour-Driven Development of Foundational UML Components.
[43] Nuseibeh, B. & Easterbrook, S. (2000) Requirements Engineering: A Roadmap. Proceedings of the Conference on the Future of Software Engineering, 35-46. http://dx.doi.org/10.1145/336512.336523. https://www.scirp.org/(S(czeh2tfqyw2orz553k1w0r45))/reference/referencespapers.aspx?referenceid=1706167
[44] Hosiaisluoma, E. (2021). ArchiMate Cookbook-Patterns & Examples. Hosiaisluoma. Finland.
[45] Togaf-Modeling (2020). Application communication diagrams. Togaf-Modeling.org. https://www.togaf-modeling.org/models/application-architecture/application-communication-diagrams.html
[46] Richardson Ch. (2014). Pattern: Microservices architecture. Available online at: http://microservices.io/patterns/microservices.html.
[47] The Open Group (2011). Sample catalogs, matrices and diagrams. The Open Group. USA. available online at: http://www.opengroup.org/bookstore/catalog/i093.htm.
[48] Pavel F. (2011). "Grid Database—Management, OGSA and Integration", Academy of Economic Studies Romania, Bucharest, Database Systems Journal, Vol. II, No. 2/2011, Romania.
[49] Javatpoint (2021). Understanding SWOT Analysis. Javatpoint. https://www.javatpoint.com/swot
[50] Jonkers, H., Band, I., &Quartel, D. (2012a). ArchiSurance Case Study. The Open Group.
[51] BizzDesign (2022). Modeling a SWOT analysis. BizzDesign. https://support.bizzdesign.com/display/knowledge/Modeling+a+SWOT+analysis
[52] Störrle, H., & Knapp, A. (2006). UML 2.0 – Tutorial/Unified Modeling Language 2.0. University of Innsbruck. Austria.
[53] The Open Group (2020). Microservices Architecture – SOA and MSA. The Open Group. https://www.opengroup.org/soa/source-book/msawp/p3.htm
[54] Trad, A., & Kalpić, D. (2020a). Using Applied Mathematical Models for Business Transformation. IGI Complete Author Book. IGI Global. USA.

[55] Peterson, S. (2011). Why it Worked: Critical Success Factors of a Financial Reform Project in Africa. Faculty Research Working Paper Series. Harvard Kennedy School.
[56] Quinlan, C. (2015). Business Research Methods. Dublin City University. Cengage Learning. Ireland.
[57] Tkachenko, E. (2015). 5 key attributes of requirements testing: Know before you code. EPAM Systems. https://techbeacon.com/app-dev-testing/5-key-attributes-requirements-testing-know-you-code
[58] OMG (2022). DECISION MODEL AND NOTATION (DMN). OMG. https://www.omg.org/dmn/