An Enterprise Architecture Framework for Integrating the Multiple Perspectives of Business Processes

Eng Chew and Michael Soanes

Department of Information Technology, University of Technology, Sydney, Australia
{Eng.Chew,Michael.G.Soanes}@uts.edu.au

Abstract. Existing business process design strategies do not address the full breadth and depth characteristics of business processes. Multiple perspectives of business process design must be supported and integrated. Enterprise architecture frameworks provide a useful context to define and categorise these multiple perspectives. Levels of abstraction of business, systems and technology represent the lifecycle phase ranging from business requirements definition through to execution. Different deliverables are relevant to each level of abstraction. The business architecture consists of a set of modeling perspectives (process, activity, resource and management) that represent types of business requirements. The technology architecture defines a classification of execution architectural styles. The systems architecture consists of a meta-model that defines the fundamental concepts underlying business requirements definition facilitating the integration of multiple modeling perspectives and mapping to multiple execution architectural styles, thereby facilitating execution of the business requirements.

Keywords: enterprise architecture framework, business process modeling perspectives.

1 Introduction

There have been multiple similar proposals (Harmon [1], Davenport [2], Soanes[3]) advocating that existing business process design strategies do not adequately support the full range of characteristics of business processes. Soanes introduced the concept of the breadth / depth complexity matrix to define the characteristics of business processes. The matrix is defined as the combination of breadth (the range of activity types from structured to ad-hoc) and depth (the range of abstraction levels from coarse to fine grained) resulting in four quadrants (structured/coarse, ad-hoc/coarse etc) as per figure 1. Soanes also proposed that a typical business process has a breadth / depth coverage that crosses multiple breadth / depth matrix quadrants. Existing business process design strategies (and their associated modeling toolsets) tend to specialise in one quadrant, resulting in incomplete (that is, parts of the process design are not formalised), or at best, fractured process designs. Un-formalised parts of the process design, results in hidden business activity that cannot be planned and controlled and more importantly, cannot be monitored, thereby reducing operational transparency and accountability.

I. Bider et al. (Eds.): BPMDS 2010 and EMMSAD 2010, LNBIP 50, pp. 133–144, 2010.

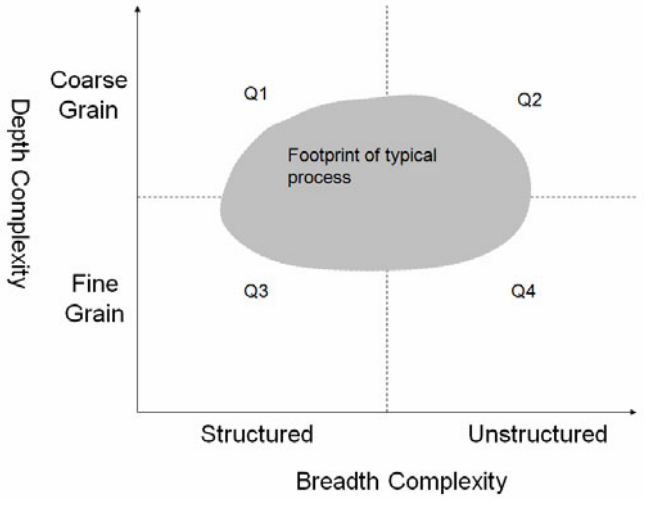

Fig. 1. Breadth / Depth Complexity Matrix

There are many stakeholders involved in business processes. However, the primary stakeholders addressed in this paper are customer, worker, resource manager (both internal and external suppliers) and business owner. Each stakeholder has a variable breadth/depth scope depending on their viewpoint (a worker has a more detailed viewpoint than a customer for example). Thus the need to support multiple overlapping business process definitions reflecting the breadth/depth scope appropriate for each stakeholder's viewpoint.

Reijers et al [4] define three dimensions (lifecycle phase, starting point and scope of improvement) to the evolution of business process designs as part of their proposed "process compass". This evolutionary view of business process design reinforces the need to support the evolution of breadth / depth scope within an ongoing iterative lifecycle of design through to implementation, with evaluation feedback from implementation to design.

Chew[5] has proposed that multiple perspectives of business processes have to be supported and integrated to address the breadth and depth characteristics of business processes. Existing process design strategies are appropriate for the scope of breadth / depth complexity they specialise in. The challenge is to identify the common underlying perspectives that facilitate integration across these multiple process design strategies, thereby supporting multiple stakeholder views of breadth / depth complexity.

The Open Group Architecture Framework (TOGAF) [6] proposes that an enterprise architecture framework (EAF) defines the dimensions and their associated deliverables as a common template for use within an architecture development method, to generate an organisation specific enterprise architecture. This paper will propose EAF as an appropriate artifact to define and categorise the multiple perspectives of business process design that are integrated and mapped across the lifecycle phases, thereby facilitating multiple stakeholder views of breadth/depth complexity. The objectives and design of meta-models as a key deliverable within the EAF will be introduced as the means of supporting the integration/mapping process. The use of

EAF's and meta-models to manage breadth/depth complexity is the key contribution of this paper.

2 Enterprise Architecture Framework Dimensions

There is no consistency across the various existing EAF's as to what are the dimensions and their associated deliverables that need to be supported. The Zachman framework [7] defines six levels of abstraction (representing the different stakeholders e.g. developer, builder etc) and six different modeling perspectives (e.g. function, data etc). The Object Management Group's (OMG's) Model Driven Architecture (MDA) [8] defines three levels of abstraction: computation independent model, platform independent model and platform specific model. TOGAF defines three levels of abstraction: business architecture, systems architecture and technology architecture. For both TOGAF and MDA, the modeling perspectives are represented by the deliverables proposed for each abstraction level. For example TOGAF defines the deliverables of the business architecture as business strategy, governance, organisation and business processes.

The Zachman framework does not attempt to integrate deliverables; a modeling perspective maps one to one through each level of abstraction. This will not satisfy the objective of integration of modeling perspectives described in the previous section. MDA has an objective of interoperability between multiple modeling notations and automating the mapping between levels of abstractions. However its scope is highly structured business domains and as such does not support the definition and integration of multiple modeling perspectives. TOGAF has recently released (Version 9.0 February 2009[9]) a meta-model to support the linking of deliverables across the levels of abstraction. The emphasis is on "linking" as distinct from "integration."

Consequently, it is proposed that these existing EAF's will not support the integration objectives defined in the previous section. It is proposed to combine terminology and concepts from existing frameworks extended with further refinements to address deficiencies identified with the existing frameworks to achieve the desired integration objective as per figure 2 below. As per Zachman and OMG's terminology, the term "levels of abstraction" defines the lifecycle phase ranging from defining business requirements through to execution. TOGAF's specific layers of business, systems and technology have been adopted as the levels of abstraction as defined in figure 2. This choice is based upon the reduced granularity of abstractions (three vs. six in Zachman) and their definitions are more aligned with the proposed deliverables (although OMG's definition of its three levels of abstraction is very similar to TOGAF's and could be adopted as an alternative terminology and definition). The term "deliverables" defines the artifacts that are produced at each level of abstraction. The term "modeling perspectives" specifically relates to a key deliverable within the business architecture that defines the different types of business requirements that need to be modeled (as per Zachman's terminology and concept). Finally the term "multiple dimensions" is proposed to be more applicable as a broader concept of the combination of levels of abstractions and the deliverables of each abstraction level (in effect the total set as defined in Figure 2). The term "stakeholder viewpoints" is proposed to represent a specific stakeholder's scope of interest of the total multiple dimension set. In Zachman, each level of abstraction is a stakeholder (eg system logic as a level of abstraction is

Level of Abstraction		Deliverables for each Level of Abstraction
Business Architecture	Definition of the business including strategy, governance, structure and processes.	Modeling perspectives
		Modeling constructs
		Modeling notations / languages
Systems Architecture	Structure of the components, their relationships and the principles of their design and evolution over time.	Mapping to Business Architecture
		Meta-model
		Mapping to Systems Architecture
Technology Architecture	The software and hardware infrastructure required to support the deployment of systems components	Execution architectural styles
		Specific execution standards and / or specific vendor products

Fig. 2. Proposed enterprise architecture framework

assigned to architects and designers). However it is proposed that it is more useful to allow stakeholders to have a scope of interest across multiple levels of abstraction and thus the need to support a separate concept of stakeholder viewpoints.

The following sections will first introduce the role and objectives of a meta-model in supporting the integration of multiple modeling perspectives and mapping them across the multiple levels of abstraction. Further detailed explanation of the deliverables in each level of abstraction is then provided.

3 Modeling Perspectives Integration and Mapping

Generally a meta-model is a collection of concepts (constructs, rules, terms) needed to build specific models (which are instantiations of the generic meta-model) within a domain of interest. Specifically within the scope of the proposed EAF, a meta-model is a definition of the fundamental concepts underlying business requirements definition facilitating the integration of multiple modeling perspectives (and their multiple modeling constructs/notations) and mapping to multiple execution architectural styles, thereby facilitating execution of the business requirements.

The Zachman framework provides little support for integration/mapping. TOGAF originally drove integration/mapping through a structured methodology rather than through integrating the deliverables of each perspective. TOGAF Version 9 [9] launched in 2009 has added a meta-model that links the artifacts together. OMG [8] have released a meta-model called Business Process Definition Meta-model (BPDM) [10] that aims to support the mapping of multiple procedural modeling notations to a Service Oriented Architecture (SOA) execution architectural style. BPDM remains

within a very structured prescriptive approach and would not address the breadth / depth coverage issue. The Super project [11] is a European Union project to implement semantic web concepts within the business process management (BPM) domain with the objective of raising BPM from the IT level to the business level. A specific goal is to integrate procedural and declarative modeling constructs through a set of meta-models (they call them ontologies). Although the most promising from an integration objective, it would still remain a very process centric approach.

Extrapolating from the proposed EAF in figure 2 and the role of the meta-model within the EAF, the following is a proposed set of six objectives of the meta-model. (1) To support the mapping of multiple modeling perspectives of the business architecture (in particular modeling constructs/notations combinations within each modeling perspective) to the meta-model facilitating the integration and conflict resolution of business requirements defined in each modeling perspective. (2) To support interoperability between modeling constructs/notations allowing the reverse mapping back from the meta-model to any modeling construct/notation retaining where appropriate the presentation semantics used in the original requirements definition construct/notation. (3)The reverse mapping from the meta-model to a modeling construct/modeling notation for presentation, should support the filtering of the business requirements to multiple levels of presentation detail reflecting the intended audience (e.g. a worker would require presentation of detailed business requirements while a manager would require presentation of high level business requirements). (4) The mapping from the meta-model to multiple execution architectural styles within the technology architecture to facilitate execution of the business requirements. (5) To support an evolution of design from execution feedback through an iterative design/execute/evaluate lifecycle. (6) The mappings to be automatable as much as possible with required manual intervention clearly defined.

4 Business Architecture

The business architecture defines the structure and operations of a business (where business processes is a subset). Within the business architecture there are three sub layers. The modeling perspective defines the type of business requirements that need to be modeled. Process, activity, resource and management are the proposed modeling perspectives as described in more detail below. Modeling constructs are conceptual approaches to modeling each business requirements perspective. For example, business processes (as a modeling perspective) can be modeled as prescriptive activity flows (as a modeling construct appropriate for highly structured aspects of processes) or declaratively as business rules (as a modeling construct appropriate for unstructured processes and exception handling aspects). Modeling notations / languages are the specific standards or vendor products that are used as implementations of different modeling constructs. For example, for prescriptive activity flow modeling as a modeling construct, there are multiple graphical notations including Business Process Modeling Notation (BPMN) [12] and Unified Modeling Language (UML) Activity Diagrams [13]. Modeling constructs can have multiple levels of abstraction as alternative approaches to addressing each modeling perspective. In the case of prescriptive activity flow: use case scenarios is a high level construct (where there are standard notations such as proposed

within UML); graphical activity flow modeling as described above is a mid level construct and procedural code is a low level construct (which has a multitude of language options).

The business architecture needs to support the modeling of multiple perspectives of the business. As described in section 2 above, existing EAF's are not consistent with their choice of modeling perspectives. Chew [5] introduced the Process / Activity / Resource / Management (PARM) framework as a model of business requirements perspectives that need to be supported and integrated to address breadth / depth complexity as follows:

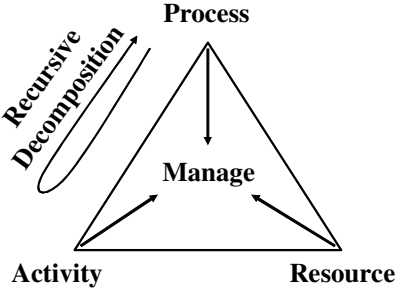

Fig. 3. PARM Framework – Modeling Perspectives

The process modeling perspective (MP) focuses on controlling, guiding and restricting the sequence of activities performed for specific process instances. Its measurable objective is to meet the customer's end to end service delivery expectations. The activity MP focuses on the facilitation of an environment to manage human activity with the recognition that human resources will prioritise their own execution of multiple activities across multiple processes simultaneously based upon their own individual work practices. Its measurable objective is to provide the most effective (both productivity and quality) environment for the completion of all work across all processes for the process or knowledge worker. The difference between a process and an activity is not definitive given that an activity can be a sub process decomposed at the next level of granularity. As a guideline rather than a rule, a process is sequencing non-contiguous activities over a long business lifecycle across multiple resources as compared to an activity, which is coordinating contiguous actions (sub-activities) within a system processing lifecycle usually by the one resource. The resource MP forecasts, plans, schedules and assigns resources to activities. Its measurable objective is to maximize the utilisation and therefore the efficiency of the total resource pool. This MP captures the resource manager's (e.g. team manager) requirements. The Management MP integrates the process, activity and resource MP's through balancing the tension between service, cost and quality expectations. It reflects the requirements of the business owner of the process.

An issue is whether PARM's choice of modeling perspectives to be integrated are complete and appropriate. In particular, social/people and information/data modeling perspectives are further candidates to be considered. As a first pass in the definition of the total strategy, the scope will be restricted to the four PARM perspectives.

5 Systems Architecture

A systems architecture defines the components of the system, their relationships and the guidelines and principles governing their design and evolution. The key deliverable within the system architecture is a meta-model which is the key enabler of modeling perspective integration and mapping across levels of abstraction as per figure 4 below.

The meta-model is positioned within a Service Component Architecture (SCA) [14] focus (which is a current in progress standard proposed as an evolution to Service Oriented Architecture (SOA)) and thus the adoption of terminology of "components" and "services" as proposed within SCA. Abstracting and encapsulating the business domain into components at multiple nested levels, facilitates management of depth complexity. Meta-model processing applies to the highest level component model and privately within each component at each nested level.

In essence, the fundamental concept of performing an individual instance of work, is the allocation of a resource to an activity that is to be performed on the process instance based upon the current state of the process instance, resulting in the transition to a new state, thereby triggering the need for further work appropriate for the new state. This cycle continues until no further work is required on that process instance. Each individual work instance exists within a total set of work across multiple processes, activities and resources where the prioritisation of each work instance is driven by cost, quality and schedule drivers of the various stakeholders (customer, worker, resource manager, business owner).

There are three key business operational logic definitions inherent in the above work definition. Firstly, defining what are valid states and what are valid and invalid transitions between these states, including where known, what is the conditional routing logic to trigger each valid state transition. Conditional routing logic can be formally defined as part of the process definition but also maybe unknown as it is embedded as part of the activity processing logic or embedded as human resource expertise. The second business operational logic definition is what activity (or activities) are to be performed for a process instance in a given current state. This current state/activities to be performed association may be formally defined (particularly for automated system activities) or maybe unknown as it is part of a human resource's role of deciding what manual activities are appropriate to perform for the process instance current state. The third business operational logic definition is defining what resource is required to perform the activity (ies) for the process instance in its current state. This current state/resource association can be formally defined allowing work to be pushed to the appropriate resource, or undefined requiring resources to pull work of the appropriate state.

Extracting from the above fundamental concept of performing work, the meta-model proposes that the triple relationship of process instance state transition (hereafter shortened to state transition), activity performed and resource usage is the fundamental conceptual structure to be modeled. Ideally all three associations (state transition/activity/resource) would be defined, however as per the above description of work, state transition is the core and associations with activity and resource are optional. Reflecting the importance of state transitions, this view of work will be called the "state based view of work" (as distinct to a traditional activity flow based view of work).

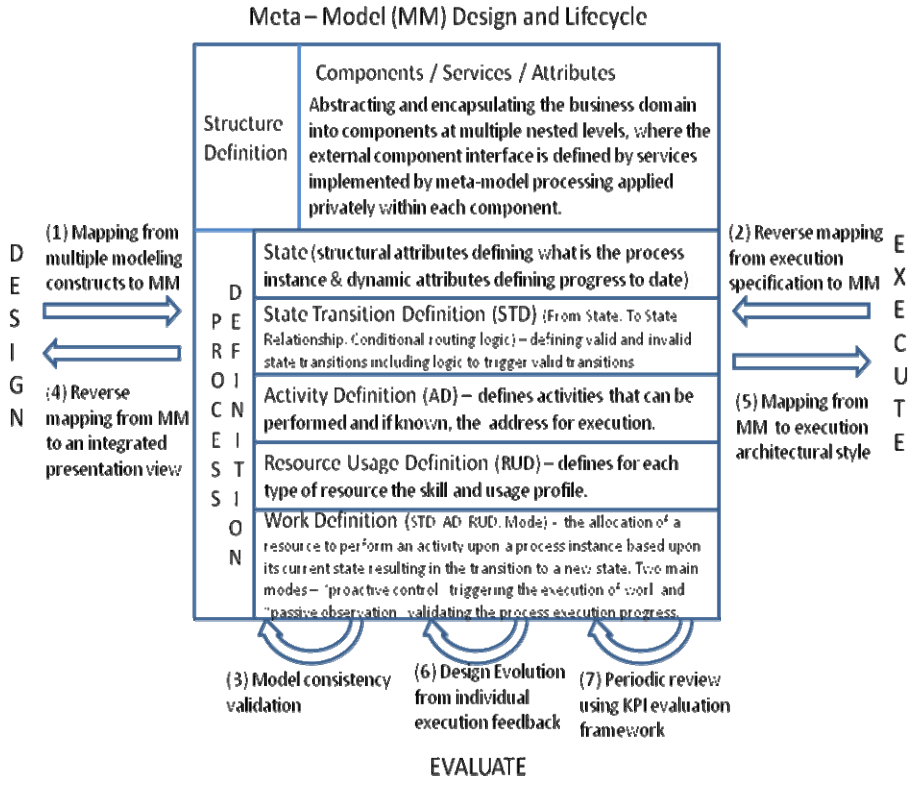

Fig. 4. Meta-Model Design and Lifecycle

A state can be any set of attributes and their values appropriate to the business domain. However there are structural and dynamic components to state definition. The structural component defines what the process instance is. A common set of attributes would be process type, client type and product type (although they would vary per business domain). They are structural in the sense that a process instance will not change these attributes frequently, if at all. The dynamic state definition component defines what has happened so far to the process instance. A common set of attributes would be business state and system state as a summary of the existing state, although detailed data attributes could be used.

There are three principles it is proposed that the work definition concept must support to assist the meta-model integration/mapping objectives. Firstly, the work definition concept through the "relationship" attribute will allow the definition of both definitive state transitions (transitions that either "must" happen, "should" happen or "can" happen) and restrictive state transitions (transitions that "should not" happen or "must not" happen). Traditional prescriptive process definition approaches do not support restrictive process definition although declarative rules type approaches would support both definitive and restrictive. Restrictive process definition is useful for defining compliance type business requirements. Secondly, the work definition

concept through the "mode" attribute will support a proactive control mode (where meta-model processing is the trigger for business process execution) and a passive observation mode (where business processing is occurring beyond the control of the meta-model). In the passive mode, the meta-model is either providing proxy services (creating defined activity events from known state transitions and vice versa) or reconciliation services (where state transition events are reconciled back to activity events and vice versa). Finally, a fundamental meta-model design feature is that the state transitions are implemented as a hierarchy of transitions where state transitions can overlap or be duplicated at multiple breadth and depth levels of the hierarchy. For example, take the simple state transition flow: A transists to B and B can transist to either C, D or E. Wild card character definition supports variable breadth definitions. For example, if B / C is the core transition and B/D and B/E are low volume exception cases, then an appropriate breadth definition could be B/C as an explicit definition and B/* to rollup all exception cases. Multiple / overlapping depth definitions would support detailed definition of A/B, B/C, B/* transitions as well as support higher level duplicated definitions of A/C and A/*. This multi-filtered hierarchical definition of state transitions is an important enabler of supporting varying breadth / depth scopes for the multiple modeling perspectives.

Having defined the design of the meta-model, the lifecycle of the meta-model will be explained. A state based view of work definition tends to be a more evolutionary definition lifecycle as represented in the iterative design / execute / evaluate lifecycle in figure 4. The ability to succinctly define in advance all possible valid state transitions is difficult and thus work definition must be defined within an ongoing feedback cycle adjusting appropriately with new or modified state transitions when execution feedback indicates that state transitions are occurring that had not been envisaged. It is proposed that this more accurately reflects the complex reality of business operations than traditional static activity flow business process design.

The mapping from multiple modeling constructs to the meta-model (lifecycle phase 1 in figure 4 above) represents the key phase of mapping from the business architecture to the system architecture. It would be an ideal capability to reverse map the processing definitions from existing legacy execution environments back into the meta-model (lifecycle phase 2). Having populated the meta-model from multiple modeling constructs and the reverse mapping from existing legacy environments, consistency validation logic (lifecycle phase 3) is necessary to analyse the meta-model to detect issues such as overlapping state transition definitions and inconsistent state transition definitions. Reverse mapping back from the meta-model to integrated modeling constructs (lifecycle phase 4) provides a consolidated view of the multiple perspectives at multiple levels of detail appropriate to the audience. After this consolidated review, the meta-model content can now be mapped to suitable execution architectures (lifecycle phase 5) representing the key phase of mapping from the system architecture to the technology architecture. As work is performed within the execution environments, event feedback will be received. There is scope to review and evolve the meta-model content based upon each work event (lifecycle phase 6). For example, where those events represent state transitions that have not been previously defined. Finally, there is an automated periodic review (lifecycle phase 7) of the existing state definitions and their breadth and depth granularity definition based upon aggregated statistical feedback of actual execution events. For example, very low frequency exception based

state transitions maybe rolled up into higher level summarised state transition definitions to avoid an unmanageable explosion in exception case definitions, whereas high volume state transitions justify detailed definition (although additionally influenced by stakeholder viewpoints). Supporting the evaluate processing is a key performance indicator (KPI) evaluation framework that measures the depth and breadth design effectiveness of the state based meta-model definition. The full definition of this KPI evaluation framework is the subject of a future paper.

6 Technology Architecture

The technology architecture defines the software and hardware infrastructure intended to support the deployment of system components. The technology architecture has two levels. Execution architectural styles define different categories of approaches to executing business requirements that in the second level may have multiple standards or vendor specific products that implement that execution style.

Many architectural styles have evolved as a means of supporting this deployment. A classification of architectural styles is proposed by Fielding [15]. Fielding bases his categorisation on the constraints inherent in the communication of components of the system. Fielding defines twenty-two styles on this basis with the recognition that there are further possible styles. Examples of Fielding's styles are client server, mobile agent, event based integration and distributed objects. It is out of scope of this paper to define in detail the multitude of architectural styles and standards in the implementation domain. The choice will be influenced by specific business domain requirements and the existing legacy implementation environment.

However, this paper proposes a simplified categorisation of architectural styles based upon process design coupling (how strong is the dependency between participants of the process) and process design cohesion (how centralised is the control of process flow) as proposed in the following diagram:

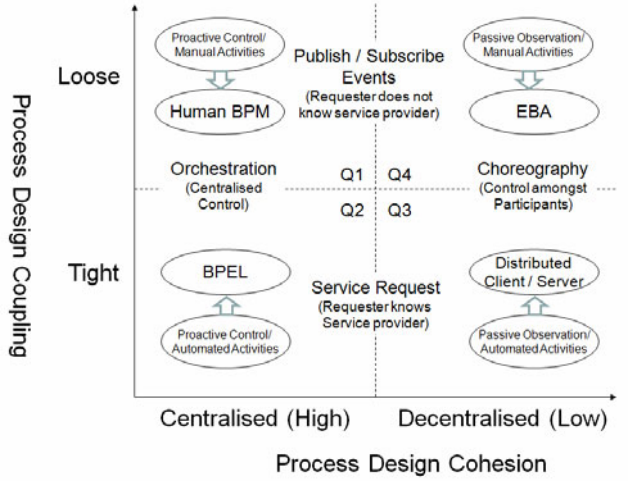

Fig. 5. Categorisation of execution architectural styles

As per the definition of the meta-model in section 5 above, it is proposed that there are two modes of meta-model operation: proactive control and passive observation. In applying these modes to an execution environment, an additional dimension of whether the activities are manual human resource executed or automated systems activities is useful to classify the appropriate execution architectural style for each mode. Thus we have four meta-model operational modes: proactive control/manual activities (meta-model facilitates process execution but requires a resource as link between process and activities); proactive control / automated activities (meta-model facilitates process execution and has direct control over activities); passive observation / automated activities (meta-model validates but does not control process execution within an environment of automated system activities) and passive observation / manual activities (meta-model validates but does not control process execution within an environment of human resources performing activities).

Although not definitive, the four meta-model operational modes map to the four quadrants of the simplified categorisation of execution architectural styles as per figure 5. Additionally, although not the only choice of example execution architectural styles for each quadrant, the following will propose example execution architectural styles that map to the four meta-model operational modes. Human based BPM is the typical solution used for highly centralised process control (through a workflow engine) with loose coupling, where the human work performers can be easily interchanged (as defined in quadrant 1). This is a suitable execution architectural style for the proactive/manual meta-model operational mode. Business Process Execution Language (BPEL) [16] has evolved as a common standard for supporting system based business process management (reflecting the history of BPEL as evolving from enterprise application integration (EAI) where message passing between applications is the key BPM requirement). BPEL is more appropriate for quadrant 2 reflecting tight coupling (client knows server) and highly centralised process coordination. This is a suitable execution architectural style for the proactive/automated meta-model operational mode. Tight coupling (client knows server) but distributed (and also evolutionary) control amongst participants as defined in quadrant 3 is typical of a distributed client server type architecture where the process evolves over the evolution of multiple interactions amongst clients and servers. This is a suitable execution architectural style for the passive/automated meta-model operational mode. Finally, the preferred architecture (as it is the most flexible) is an event based architecture (EBA) where control is decentralised amongst participants and where communication between participants occurs via events that are published and interested participants subscribe (as defined in quadrant 4). The mapping of example architectural styles to the specific quadrants does not negate that each style can additionally (but maybe not ideally) address meta-model operational modes mapped to the other quadrants. In particular, it is proposed that EBA is suitable for all meta-model operational modes and additionally is useful as a means of integrating process execution across multiple execution architectural styles.

7 Conclusion

EAF's provide a useful context to define and categorise the multiple perspectives of business processes necessary to support the full breadth and depth characteristics of

business processes. A specific enterprise architecture framework was proposed, that defined the levels of abstraction representing the lifecycle phase from business requirements to execution. Deliverables at each level of abstraction were defined. Specifically a set of modeling perspectives were proposed (representing the types of business requirements) for the business architecture. A meta-model was proposed for the systems architecture. A categorisation of execution architectural styles was proposed for the technology architecture. The use of EAF's and meta-models to manage breadth/depth complexity is the key contribution of this paper

It was beyond the scope of this paper to define the mappings between each level of abstraction which will be future work. The total solution will then be applied to common business process scenarios that have a mixture of breadth / depth complexity profiles using a case study as real examples of each scenario.

References

1. Harmon, P.: Task Complexity Continuum. BPTrends (July 18th, 2006) published at, http://www.bptrends.com
2. Davenport, T.H.: Thinking for a Living., p. 27. Harvard Business School Press, Boston (2005)
3. Soanes, M.G.: Process Design Strategies to Address Breadth and Depth Complexity. In: Dustdar, S., Fiadeiro, J.L., Sheth, A.P. (eds.) BPM 2006. LNCS, vol. 4102. Springer, Heidelberg (2006)
4. Reijers, H.A., Mansar, S.L., Rosemann, M.: From the Editors: Introduction and a Compass for Business Process Design. Information Systems Management 25, 299–301
5. Chew, E., Hawryszkiewycz, I., Soanes, M.: Value Configuration Design – an evolution in adequate business process design. In: 8th Workshop on Business Process Modeling, Development and Support at CAISE 2007, Trondheim Norway (June 2007)
6. TOGAF accessed at, http://www.opengroup.org/togaf
7. Zachman Framework accessed at, http://www.zachmaninternational.com
8. Model Driven Architecture accessed at, http://www.omg.org/mda
9. TOGAF Version 9 accessed at, http://www.opengroup.org/togaf
10. Business Process definition Meta-model accessed at, http://www.omg.org
11. Semantics Utilised for Process Management within and between Enterprises (SUPER) project accessed at, http://www.ip-super.org
12. Business Process Modeling Notation accessed at, http://www.bpmn.org
13. UML Activity diagrams as part of UML 2.0 at, http://www.uml.org
14. Service Component Architecture accessed at, http://www.osoa.org/
15. Fielding, R.T.: Architectural Styles and the Design of Network-based Software Architectures. Doctoral dissertation, University of California, Irvine (2000)
16. Business Process Execution Language (BPEL) accessed at, http://www.oasis-open.org