SOA and Large Scale and Complex Enterprise Transformation

Mansour Kavianpour

Executive Architect, Unisys Corp USA
Mansour.Kavianpour@Unisys.com

Abstract. Service-oriented architecture (SOA) is an architectural approach to development that turns traditional techniques upside down. SOA encourages organizations to think in terms of actual business services and the associated data, rather than low level technology details. Instead of developing applications from the ground up, SOA frees organizations to start with high level business definitions for data, interfaces, documents, and processes. SOA then maps these high level service definitions onto new or existing infrastructure, regardless of the details, location, or programming language in which the systems were written[1],[2].

In this paper we share our practical experience regarding application of SOA to a very large and complex enterprise transformation. By transformation we mean modernization of legacy applications, operating systems, server components, development of new applications, and business process automation with incremental deployment option using either a traditional distributed heterogeneous environment or a set of Virtual machines deployed on a set of utility computing platforms (on-demand computing).

Transforming a large and complex enterprise requires digital visibility into the holistic view of the enterprise. This holistic view is captured as a set of interrelated models. Models are digital representations of the enterprise business architecture, the associated technology architecture, and their semantic dependencies. Models are stored in a living and manageable repository with impact analysis capability to accommodate for SOA modernization to be driven by the business needs.

We describe our 3D Visible Enterprise (3D-VE)[3] modeling methodology as the analysis phase of our SOA approach. We then describe how such analysis guided us through modernization styles, where each style prescribes the transformation of a legacy entity into its modernized form while following our SOA governance. In this approach the modernization requirements are mapped into the appropriate transformation styles and finally to technical implementation. The mapping follows our SOA governance, a set of guidelines

[1] Principles of SOA Design, A whitepaper from Cape Clear Software Inc.
[2] Best Practices for SOA with Cape Clear ESB.
[3] 3D VE (3 Dimensional Visible Enterprise) is the Unisys modeling approach to creating a more visible enterprise. It's a proven business and systems modeling framework and methodology that integrates business vision with IT execution to create organizational visibility (www.unisys.com).

B. Krämer, K.-J. Lin, and P. Narasimhan (Eds.): ICSOC 2007, LNCS 4749, pp. 530–545, 2007.

regarding transformation style selection, and the SOA design & run time governance.

Relevant SOA standards and products supporting modernization implementation are used to carry the implementation. In particular, some aspect of modernization approach and Unisys SOA governance are described as well.

In this paper we tried to describe three important areas of our overall SOA solution methodology, namely the 3D VE modeling, the Architecture Driven Modernization and applied SOA governance to the ADM style. Our plan is to publish subsequent papers each describing the details of our SOA solution methodology, specifically the modeling phase, the tools and methods used to implement the applied ADM styles, criteria selecting ADM style, the SOA governance and the associated SOA standards and products.

The paper concludes with lessons learned through such a complex transformation, especially the importance of the front-end business process analysis leading us to identify the components or a subset of the enterprise computing environment for systematic and incremental SOA transformation. Finally we discuss some of the pros and cons of the transformation applying SOA.

1 Introduction

To protect our contractual obligation we would like to avoid to reveal the name of our client organization instead we use the term “the enterprise”. The enterprise subject to the transformation is a multi billion dollar global pharmaceutical company with distributed IT infrastructures supporting its daily businesses. A modest inventory of such a complex environment includes more than 600 applications, over 3000 window stations, 3000 UNIX stations, 10s of VAX machines, mainframe computers and numerous data sources, middlewares, massive storage and network installations, proprietary securities, with software-enforced US government policies, and regulations with complex existing firewalls, numerous business and employee portals, web and complex financial applications.

The transformation asks for replacement of old computers, consolidation of applications, data sources, and middleware into a secure SOA environment. The transformation further asks for enterprise architecture with a deployment option on an on-demand computing platform using virtualization technology. The overall transformation will take 5 years with multiple phases.

Transformation can not be performed in isolation. For such a large and complex enterprise the transformation requires digital visibility into the holistic view of the enterprise. This holistic view a) allows systematic identification of hosted applications, and b) facilitates for business impact and risk analysis of the hosted applications running on the retired VAX machine. Similarly other platforms like MVS, HP-3000, AS400 and the like will eventually require to be modernized and therefore their business impact must be known before any modernization activities. Similar to a large scale, distributed software development where the design phase plays the critical role in success of the project, capturing holistic views of the existing enterprise for analysis is the key to the successful transformation. It is this level of

visibility that allows for systematic impact analysis, modernization and consolidation planning, the associated risk assessment and mitigation and project planning. Section Unisys 3D VE Methodology briefly introduces the Unisys 3D VE transformation methodology. This section sets the stage for our next step in our road map, the actual modernization.

Once the elements of the enterprise are selected for the modernization, the modernization itself requires a proven method. Unisys adopted Architecture Driven Modernization (ADM) which is an IT Modernization discipline using a model-driven approach. Unisys ADM is based on the Object Management Group (OMG) Model Driven Architecture (MDA)[4]. We apply this method to each step of transformation to implement the required modernization. Section Architecture Driven Modernization briefly describes our ADM transformation approach.

Our target enterprise architecture must address some challenging requirements. For example the client wanted an agile architecture to allow non intrusive application replacement, integration, data migration, application modernization, server modernization, and middleware and database consolidation. We adopt SOA with supporting governance. Without architecture governance ad-hoc transformation will soon create a costly IT chaos. Section SOA Architecture and Governance briefly describes the SOA governance we applied to each transformation styles.

The final SOA environment accommodates for two radically different deployment architectures, namely a traditional distributed network computing, or an on-demand Real Time Infrastructure (RTI) computing platform. The Unisys RTI approach is not discussed in this paper. We believe, due to its complex nature, a separate paper should be allocated to this topic. It is worth to mention that our RTI approach takes the on-demand computing to a new level where the demand for resources are detected at runtime and dynamically allocated.

Section Conclusion highlights the lesson we learned during the design and the description of the road map of the transformation. Some specific application of SOA technology that helped us during the transformation design is highlighted as well.

2 Unisys 3D VE Methodology

Transforming large and complex environments require modeling and blueprinting of enterprise subject to transformation in form of digital models that provide risk free, predictable, repeatable and cost effective transformation.

The 3D Visible Enterprise (3D-VE) is the Unisys modeling approach. 3D-VE makes visible the relationships between the business and the technology that supports it. It reveals the connections business strategy, business process, infrastructure and traceability. It shows how infrastructure applications, hardware and management processes work together. And it anticipates the results through impact analysis and "what-if" simulation of proposed changes.

[4] Model Driven Architecture (MDA) is a framework based on the Unified Modeling Language (UML) and other industry standards for visualizing, storing, and exchanging software designs and models. MDA promotes the creation of machine-readable, highly abstract models that are developed independently of the implementation technology and stored in standardized repositories.

Our methodology takes a holistic view across all dimensions of an enterprise. This holistic view includes modeling of Enterprise's *Business Architecture* and *Technology Architecture*. Modeling the business architecture includes modeling the Enterprise's Business Strategy and Business Process while modeling the technology architecture includes modeling of the Enterprise Applications and Infrastructure components.

Blueprinting provides a unique, four layered structure that reveals the complex relationships between business strategies, business processes, applications, and the IT infrastructure. By making these relationships visible, a high degree of traceability is achieved — making response to change a reality. This approach enables capture of Enterprise's' organization knowledge and end-to-end dependencies in a number of artifacts which continuously maintained in a Repository.

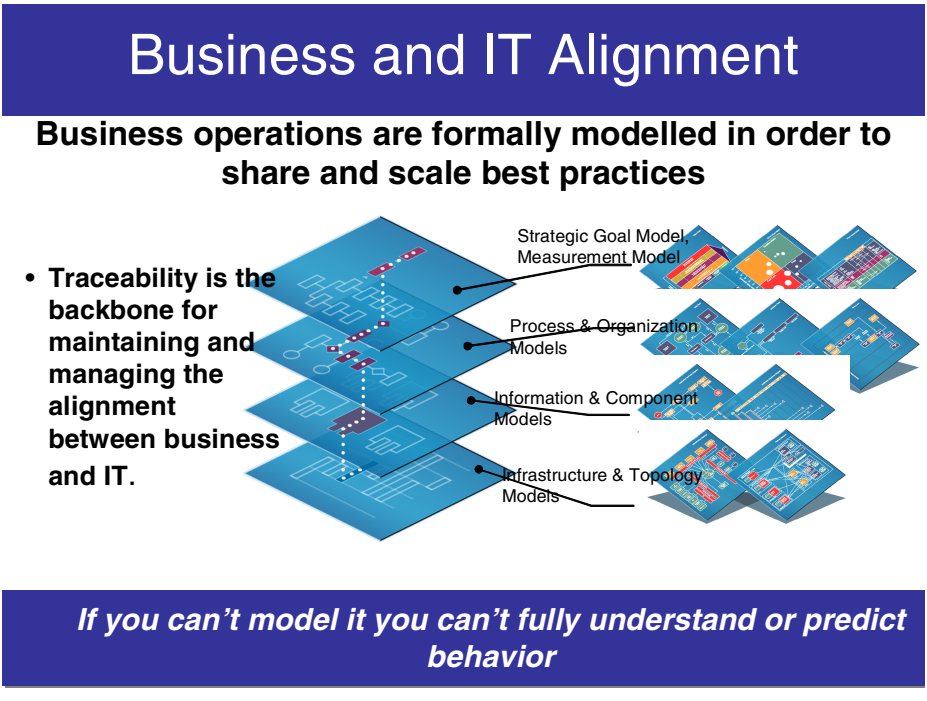
The processes involved in building the relevant Blueprints (aka knowledge repository) take two paths. The first path captures the Enterprise's business models in a number of logically related layers using existing documentations and SMEs. These layers, as defined below, are abstracting the Enterprise overall operations. Artifact models specific to each layer will be developed using Unisys 3D Blueprinting tools. A quick summary of sample Blueprints are given below. These models are all developed using ProVision[5].

- **Business Visions and Operations Models** -- Layer1 artifacts: At this level we capture the Enterprise's organizations, goals and business opportunity models. The data is gathered using organization charts, company goals and interview with management.
- **Business Process Models** – Layer2 artifacts: At this level we capture the business processes fulfilling organizations', goals' and opportunities' captured in Layer1. These models are captured as Business Swim Lanes and the Business Interaction Model. The data regarding business processes are gathered via interview with business people and subject mater expert.
- **Functional and Application Models** – Layer3 artifacts: At this level we capture Business Use Case Model, Cost Model, and Deployment Model related to business processes capture in Layer2. In most cases we discovered that business use cases for identified processes must be developed from scratch. Similarly we have to develop the cost model and deployment model. The data for creating these models usually do not exits and has to be developed.
- **Infrastructure Models** – Layer4 artifacts: At this level we capture Infrastructure Service Architecture Model, Infrastructure Service Usage Specification model supporting IT components captured in Layer3. These models capture application and node topology (the actual IT components topology). The data required to develop this model is gathered using our network agent toolset. Metadata about applications, their locations, IP address of servers, etc. are all gathered and used as input to develop the infrastructure models.

These models once are captured are the foundation for delivering an intelligent infrastructure vision. They are used to abstract and represent data and metadata

[5] We have customized Proforma Provision modeling tool with Unisys Meta schema to allow the development of 3D VE models.

describing the various IT components, their dependencies and relationships needed for transformation as well as a knowledgebase to realize a real-time infrastructure. The models are kept and managed in a repository. A graphical representation of the 3D VE is illustrated in the following diagram, the layer and the dependencies are high lighted.



Models were used for impact analyses. For example, the organization artifact model (organization model captured at layer1) is traced to one or more associated business processes (business process swimlane captured in layer2). The dependency implies that the Enterprise organization uses certain business processes. Similarly, each business process will be traced to its business use cases (business use case model captured in layre3), and finally each business use case artifact will be traced to application and computing node (Infrastructure Service Architecture Model captured in layer4).

We needed this level of visibility in order to decide on partitioning and isolating the legacy entities to be transformed into the new SOA environment. This living and maintainable repository of models and multi dimension traceability provided a complete enterprise view to the subject Enterprise business operation.

We used the business process workflow analysis (a top down analysis) to improve the existing business processes, identify processes creating backlogs, identify unwanted processes, etc. These activities are ProVision specific and are not discussed here. Once improvement identified, we used the Impact Analysis tool to identify risks,

formulate risk mitigation, estimate cost before starting the modernization project. We used Impact Analysis for removing server, or application(s) to assess and measure the impact on the Enterprise's business operation (a bottom up analysis).

In summary, the repository of models helped us with modernization implementation planning, prioritizing and planning for retirement of the existing components, consolidation, and preparation of knowledge extraction, and application of our Architecture Driven Modernization (ADM) aka Enterprise Modernization methodology.

The captured as-is models are enhanced with more related data during modernization analysis phase. We applied our ADM methodology to actually make the implementation decision which transforms a legacy entity into the desired target SOA component following our SOA governance.

In summery the repository of models, among others, provided numerous advantages. For example:

- Better strategic decision making for new modernization projects such as a legacy application replacement, server, database, and middleware consolidation.
- Helping with phasing and incremental transformation planning.
- Allowed the organization to identify, across the whole portfolio; exactly what processes, business rules, and application code are impacted by a change (market, legislative or otherwise). A process we used to identify and select applications subject to modernization, the respective project plan, risk and risk mitigation plan associated with the modernization.
- The repository became a key business asset, maintained and upgraded just as key operational and decision support systems are.
- It used as the source for Business Process Improvement and automation.
- It enabled for systematic identification of data sources, applications, external systems used internally or externally collectively called "touch points" to help with integration architecture, estimating and managing integration cost, risks and overall project management.
- Used as a secure repository for generating reports for government regulatory such as Sarbanes-Oxley.
- Used as a decision making for the overall consolidation.

Next section takes us into the ADM method and describes how we applied ADM to each entity subject to modernization.

3 Architecture Driven Modernization

Management of enterprise critical business knowledge starts with capture of the business architecture/organization knowledge in form of several models – both the current architecture and the new one as it is created. Business models in support of the model driven approach are the best way to do this. As modernization process starts and a particular process and its corresponding application(s) are selected, the

impact to the existing business model and propagation of such change must be analyzed. Model-driven approach supports automated forward engineering of the business rules into services that can ultimately be hosted and executed in a .NET framework, a J2EE app server, a Web server, an ORB server, a JVM, or any proprietary runtime environments.

Unisys IT Modernization Framework is based on the Object Management Group (OMG) MDA standard. It is a framework that helps to define and analyze major enterprise transformation styles. Each transformation style requires Knowledge Mining and Abstraction (KMA). We used the ADM (1 + 5)[6] mutually complimentary transformation styles to address the enterprise modernization requirements. In this approach the modernization requirements are mapped to the appropriate ADM transformation styles and finally to technical implementation. Relevant standards and products addressing modernization requirements are used but not discussed here. The Technical Implementation is actually the fruit of the ADM analysis which helped us to formulate standard SOA solutions based on available widely used vendor products.

Unisys IT Modernization Framework

Enterprise IT Modernization includes understanding, monitoring, maintaining, upgrading and replacement of the existing Enterprise applications. It relies on mining knowledge from existing applications and its abstraction to the level required for the specific modernization project. All IT Modernization styles utilize the outcome of the KMA effort in one way or another. The IT Modernization Framework is organized along two dimensions:

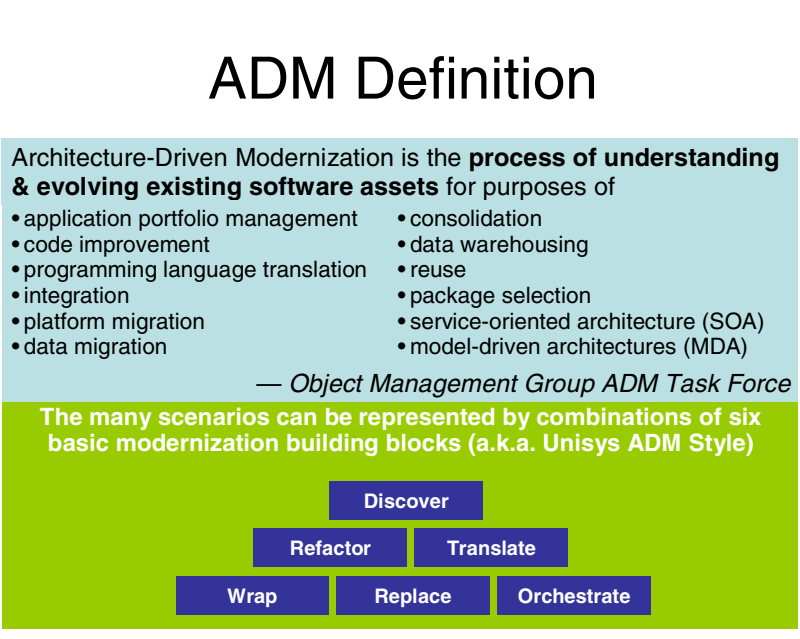
1- Scenario – a scenario is a distinct type of IT Modernization effort. The major IT transformation styles, also known as 1 + 5 ADM building blocks are:
   - One
     - Discovery
   - Five
     - Refactoring/Consolidation
     - Translation/Porting
     - Wrapping
     - Replacement – Redesign/COTS
     - Orchestration

2- Purview – a purview is a collection of artifacts at a given level of abstraction. All ADM building blocks involve effort in Knowledge Mining and Abstraction (KMA).

   The outcome of the KMA building block is a set of artifacts in the form of a model or a less formal description of the existing application at required level of abstraction. These artifacts are used as an input for each of other 5 ADM building blocks. These models also provide a single point of maintenance, a place that captures the business rules in business-like structured English and business processes in easily readable formats.

The following diagram gives an abstract view of the Unisys ADM Framework.

[6] They are (Discover) + (Refactor, Translate, Wrap, Replace and Orchestrate).



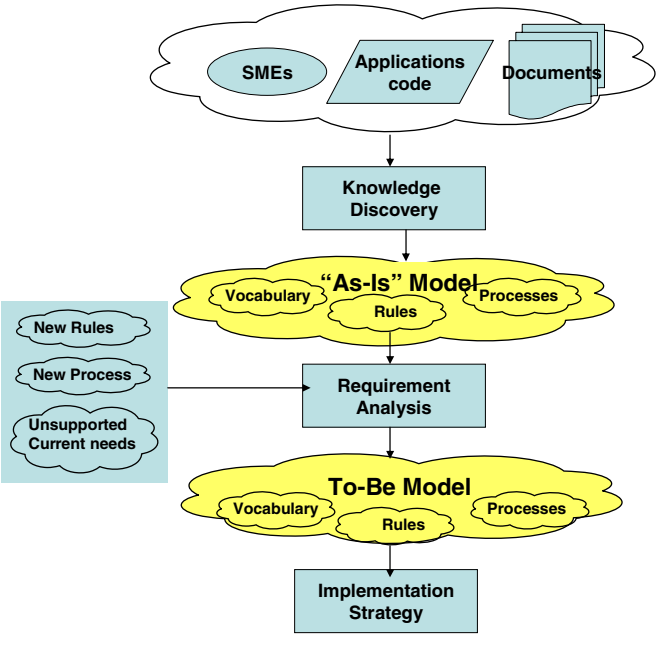
The ADM building blocks are orthogonal in the sense that they require different models produced as a result of the KMA effort and are complimentary. For example, knowledge discovery process may apply to business applications running on the VAX Open VMS, while Orchestration applies to ready-to-go application services. These building blocks serve as implementation selection category for our Enterprise modernization.

The ADM building blocks allow composing styles with the following common characteristics - at the beginning of scenario execution, the existing Enterprise's applications are analyzed and at the end – a target application is created. The target application replaces the existing application and satisfies the same (or enhanced/ modified) requirements as the existing application.

Styles or their combinations may apply to any modernization. For example, a typical modernization may require *Discovery* of existing environment and applications, *Refactoring/Consolidation* of existing applications (packaged app or custom app), *Translation/Porting* of an old system, *Wrapping* of legacy systems, *Replacement/ Redesign* of legacy components with some COTS, exposing existing and new applications as services and *Orchestration* of these services, or combination of the above. Our SOA governance has been used as an enforceable set of guidelines for development of the target application in each scenario.

The detail of "how" each ADM style are implemented will be published in future and do to space limitation are not discussed in this paper. Instead the high level descriptions of each transformation style are introduced below. Due to diversity of different Enterprise Modernization requirements, Unisys has established partnerships with a large number of translation and migration tool vendors. We purposely avoid naming any vendor in this paper.

Discovery – Remember the 3D VE analysis helped us to pinpoint the application components of the old architecture subject to modernization. This process as described above is driven by the business needs. The Discovery phase is about analysis of application component subject to modernization. The following diagram shows the high level process using Unisys Rule Modeler[7].



The "As-Is" Model will be extracted from the legacy application. The source for knowledge discovery is the existing application source code, existing documentation as well as SME (if is available). The main information captured at this level is the existing business processes, the existing business rules and the existing business vocabulary. A business process may include one or more automated and manual activities to perform a business function (e.g., fulfill an Order). A business rule specifies conditions/terms which must be satisfied (e.g., government legislation, etc). A vocabulary represents a business term such as "employee", "address", "salary", etc. This data elements form the As-Is model.

In next step, the As-Is Model will be augmented with new requirements to produce the To-Be Model. This activity usually includes client requirements analysis, addition of new rules, new business processes and the associated vocabularies. Models are kept and managed in a technology independent abstract representation (not shown in the diagram). Solution Modeler captures these in an Abstract Syntax Tree Metamodel (ASTM – defined by OMG). The abstract representation is used by proper tools

[7] Unisys Rules modeler is an integrated part of Unisys Solution modeler capable of extracting business rules, business terms and business vocabulary from the legacy application components.

(not discussed here) for generation of application component (modernized app) as well as generation of test scripts. For example, the discovered business Vocabulary is used to automatically generate the required database schema. The business rules and processes are used to generate the application code. This step requires Unisys Rules/Solution Modeler and Unisys consultant to perform the Discovery task. Rules Modeler and details of transformation are not discussed due to space limitation in this paper. In average we experienced 80% of the process above driven by the tool (automated) while 20% required manual activities.

Refactoring/Consolidation -- This style includes all types of existing application improvements such as rewriting data definitions, removing data redundancies, data consolidation and migration to relational databases, code streamlining, removing code redundancies, performance improvements, etc. Refactoring does not include improvements which significantly enhance or modify the set of requirements. Modern translation technique, including OMG-like Knowledge Mining and Abstraction techniques are used to semi-automate the Refactoring process. For example, a code fragment representing a sub-tree structure of an Abstract Syntax Tree (AST) generated during discovery can be map to re-factored code and new re-factored code can be generated using the same AST. There are few vendor tools that allow re-factoring and re-architecting to be exercised at AST level before generating the target code. Some advanced tools provide rule based AST generation. The rules include code patterns that need to be defined manually. We found AST to AST re-factoring much efficient than code level re-factoring. Code level re-factoring complicates the version management and maintenance.

Enforced SOA Governance -- The final re-factored application architecture follows Unisys SOA Design Time Application Governance.

Translation/Port -- This building block involves automated or semi-automated porting of the existing system, packaged or custom applications to the new platform or its translation to a modern (usually object-oriented) language. In most legacy modernization vendor products and tools are used to perform automatic translation. Unisys has specified a number of translation tools. Based on application implementation language and operating systems, specific tools will be used for translation. This scenario well suited for applications where direct translation results in better ROI. This scenario applied to all applications running on the VAX VMS platforms. Almost all translated apps needed re-factoring. There are few vendor tools that includes modern compiler technique in generation of language parser, rule based AST generation, and complete separation of data migration from code migration. We found rule based AST translation much more efficient than direct code translation. In direct code translation data and code migration can not be separated due to one-on-one dependency of code to data at transformation time.

Enforced SOA Governance -- The final Translated application architecture follows Unisys SOA Design Time Application Governance.

Wrapping -- This building block breaks the existing monolithic application into multiple parts, each represented as service using SOA techniques (mostly exposing the parts as Web Services). This style is an essential part of one of the most occurring

styles used in most modernization and is used for those applications where the Enterprise has made significant investment. SOA tool vendors almost all provide support for wrapping of applications written in .NET and J2EE. The real challenge is legacy languages. Very few SOA vendors provide IDE environment for wrapping legacy components written in languages like COBOL, C, C++ and others. Selection of right SOA tools plays important role in success of transformation. Wrapping techniques exist for exposing CORBA, COBOL, .NET, J2EE, Java, and C++ components as Web Services.

Enforced SOA Governance -- The final application architecture follows Unisys SOA Design Time Application Governance.

Replacement – Redesign/COTS -- This style includes a comprehensive transformation using Unisys Solution Modeler application generation. Note that To-Be Model is a technology-independent model. For this style of transformation we will perform feature gap analysis verses packaged or COTS products. If match found we will use the COTS app, if match not found and cost analysis (not discussed here) suggest to develop a new application, we then use our solution modeler application generation. Applications can be generated for either .NET or J2EE platform. The gap analysis is a great exercise (we used Rational RequisitePro for gap analysis) for specifying configuration parameters if COTS component if we decide to use COTS. We found rule based AST translation approach is the best fit when COTS component replacing a large portion of the existing legacy applications. The interface to the COTS component is modeled in the translation rules, providing semantically correct AST generation as well as final target language code generation.

Enforced SOA Governance -- The final application architecture must follow Unisys SOA Design Time Application Governance. The COTS selection follows Unisys SOA COTS selection guidelines.

Orchestration – Applying any style of transformation discussed above finally produces a modernized component adhering to Web Services standard architecture, i.e., Service components with WSDL as its interface. This orchestration assumes that the To-Be Model has already been developed. Part of the To-Be Model includes the To-Be processes. Solution Molder is used to generate the corresponding orchestration in BPEL (Business Process Execution Language) standards. BPEL script is loaded into a COTS orchestration engine. The BPEL engine finally executes the orchestration, according to the business processes defined in the to-be Model, while invoking the modernized components using their corresponding WSDL interface.

The orchestration may include other components such as security interceptors, transformation components, auditing and more. This style is an essential part of the modernization with requirement for business process automation. It is applied once and maintained after.

Enforced SOA Governance -- Orchestration is a native architectural aspect of Unisys SOA Governance which follows the industry standard, i.e., Web Services and other related open standards. BPEL applies to those application components that have already confirmed to SOA Governance, i.e., they are transformed into first class Web Service.

As we have stated in the introduction to this paper, the detail of transformation, the technique and tooling, and the translation processes deserve a complete separate paper. We plan to publish subsequent papers regarding the overall transformation and SOA techniques used in this project.

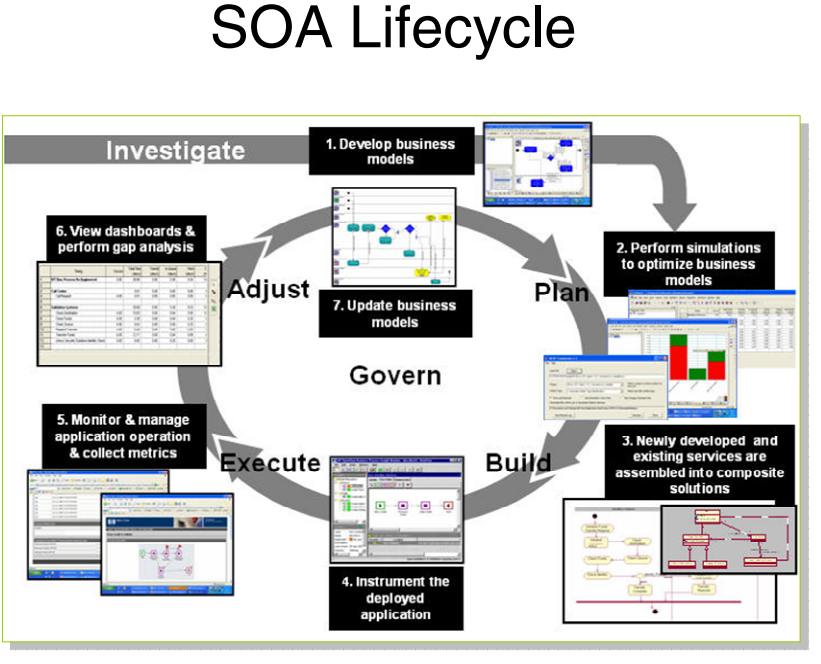
We now very briefly describes our SOA Governance, a set of guidelines helped us during transformation, i.e., design and implementation of services, service policy and runtime governance. This is indeed an introduction only. We plan to publish a separate paper discussing for example how we deploy the Enterprise Service Bus to manage runtime policy and governance.

At the time of writing this paper we have completed a successful pilot project to demonstrate our overall Enterprise Modernization methodology, tools and technique. Large scale modernization planned to follow.

4 SOA Architecture and Governance

The Enterprise Modernization style briefly described in the Unisys MDA section form the mechanisms for our transformation. These mechanisms followed the Unisys SOA Governance. Our governance includes SOA reference models, the corresponding SOA reference architectures and our SOA maturity model (none discussed in this paper).

The scenario driven (ADM) uses this governance guidelines during implementation. The following diagram summarizes our Governance.



Our SOA Governance is an agile and efficient decision and accountability framework that provides the capability to organize, understand, and manage SOA information to govern the planning, building and managing of a SOA system. In support of this goal, our SOA governance works to enable the success of the following key assets in delivering SOA:

- Prioritized enterprise and business unit service roadmaps
- Service lifecycle (specification, design, development, deployment, operation)
- Best practices, standards selection, and enforcement
- Funding models, financial metrics
- Reference architecture
- Change management to control services and registry sprawl

The governance also enables and tracks the following cultural and team transformation activities:

- Education and skills development for both business and technology groups
- Organization structure alignment for SOA, including clear roles and responsibilities for stakeholders
- Removal of organizational issues around joint ownership of processes and assets, clarified communication channels, and processes
- Enablement of successful cross-functional collaboration throughout the service lifecycle
- Individual incentive and measurement changes for SOA

It's important to establish both the organizational aspects of SOA governance, such as those outlined above, concurrent with technology enablers to enforce governance at both design and runtime. The technology aspect of the governance includes Design time and Runtime governance.

From a run-time aspect, there are requirements such as service level agreements (SLAs), routing, transformation and security that require different infrastructure elements for enforcement. Some of the technology components required to enforce Runtime Governance listed below:

- Guideline for SOA product selection to support Runtime management and monitoring of policies and SLA's.
- Repositories for version control, change management, impact analysis
- Registries for bridging of heterogeneous design-time environments to runtime infrastructure, controlled provisioning of services, and associations of runtime policies to provisioned services
- Management of contract metadata and enforcement of service contracts
- Messaging intermediary for enforcement of runtime policies such as routing, transformation, SLA, and security policies
- SOA management for runtime SLA enforcement and gathering of metrics for evolution to the next level of SOA maturity

From a design & implementation time governance aspect, the following governance is enforced:

- **Design Governance**
  - 3D VE guidelines for capturing business and technology architectures of as-is models. Impact analysis to guide through selection of the existing IT elements for modernization.
  - ADM guidelines for Knowledge Mining and Abstraction (KMA) of the identified IT elements.
  - ADM guidelines for discovery and scenario selections and analysis of the identified IT elements.
  - Standard Tools to support lifecycle processes -- capture of as-is models, and capture of to-be models.
  - Business service governance
- **Implementation Governance**
  - Guideline for use of SOA IDE (pluggable to Eclipse)
  - Guidelines for service granularity
  - Design Guideline for exposing legacy components, legacy data, legacy applications
  - Guideline for building services and orchestrations
  - Guideline for Service Registry
  - Guideline for Test the services and orchestrations
  - Guideline for deploying services and orchestrations
  - Standard Repositories of service assets and all the associated documentations that goes along with it
  - Guideline for Quality assurance
  - Guideline for Discovery mechanism for service consumers and orchestrations

And finally from the Management aspect the following governance is enforced:

- A distributed secure SOA console providing features such as:
  - Policy Management
  - Service security management
  - Service Registry
  - Service configuration and customization
  - Protocol management
  - Service Quality Assurance and Validation

Additional technologies were also utilized to facilitate non-policy facets around SOA governance. Some of the technologies considered are:

- Portals for centralized dissemination of SOA information and access control of SOA assets such as reference architecture documentation
- Dashboards for graphical representation of SOA metrics
- Business intelligence to enable SOA metrics trend analysis and scenario forecasting
- Workflow to automate SOA governance processes and enable quality and control gateways
- Service and project portfolio management to enable a holistic and more informed decision process around service candidate selection, service versioning, service retirement, and SOA investment decisions.

We found the governance a necessary element for the successful and consistent transformation into the SOA. Adhering to the underlying principle of SOA, SOA governance spans both organizational and technical boundaries. It is the critical element to enabling an organization to successfully manage and control the cross-divisional, distributed nature of SOA. Successful SOA governance ensures that an organization is prepared to respond to changing market requirements in a more agile manner. This is dependent upon the establishment and enforcement of SOA organization and governance practices via all elements outlined above—structure, process, and technology.

5 Conclusion

Having access to holistic view of the enterprise provided us with systematic but business driven transformation selection. Gathering holistic views of the enterprise as a set of interrelated models helped us with modernization implementation planning, prioritizing and planning for retirement of the existing components, consolidation, and preparation of knowledge extraction, and application of ADM. Prioritizing implicitly help with incremental modernization. Legacy components are modernized in a logical order as defined in the captured 3D VE models.

We believe ad-hoc selection of old application components without knowing the business impacts will cause the SOA modernization to fail. ROI is a vital part of large scale modernization and having a systematic methodology to help with selecting, prioritizing and planning of the old IT touch points for modernization significantly reduces the risks while guarantee the SOA project to succeed.

In a large and complex environment without a transformation framework like MDA, the modernization will become almost impossible. We believe ad-hoc selection of old application components, with ad-hoc SOA implementation strategy for the required transformation will hardly succeed. MDA styles guided with supporting SOA governance are the key to success of large scale SOA transformation. The governance specific to each styles removes any ambiguities regarding implementation approach.

SOA governance spans both organizational and technical boundaries. It is the critical element to enabling an organization to successfully manage and control the cross-divisional, distributed nature of SOA. Successful SOA governance ensures that an organization is prepared to respond to changing market requirements in a more agile manner. This is dependent upon the establishment and enforcement of SOA organization and governance practices via all elements outlined above—structure, process, and technology.

SOA like any other technological innovation requires proper skills. SOA implicitly must meet a set of challenging requirements, and that is the integration and interoperation of disparate applications, systems, databases, middlewares, and more. This vast area of integration technology opens numerous opportunities for SOA vendors to inject their own proprietary, none-interoperable features. Uneducated selection of SOA vendor products could quickly produce yet another proprietary (legacy) environment! Likely numerous Web Services standards have been produced by standard bodies and implemented by a number of SOA vendors. To move into real

SOA environment, we strongly recommend the use of SOA products that adhere to the open standards. For example an Enterprise Service Bus without using XML standards, the de-facto messaging, routing, transformation, and security mechanisms and more, will fall into EAI category, where each EAI hub used to introduce their own proprietary messaging scarifying interoperability and therefore orchestration. Service granularity, design time decisions, runtime policies must some how be defined before any service being designed and implemented. This is where SOA Governance plays a key role in success of solid design, test, deployment and maintenance of a large SOA environment.