A REVIEW ON ARCHITECTURE DRIVEN MODERNIZATION - ADM

AMIT WASUKAR, RITESH DESHMUKH, ANUJA NEWASKAR

Abstract: Today there are things which gets older but as it gets older, the importance of these thing also increases. The reason for this is the operations they are capable to perform. In todays scenario, the need for organizations to use their old systems called as legacy systems is rapidly increasing, to cop-up with this situation there must be some way to deal it. So, one solution that is available is the modernization of the existing systems, which ultimately helps in the proper usage of existing systems. The concept which hepls us to understand and makes system reusable is the "Architecture Driven Modernization", through this we can transform the architecture of the existing system to the new one as per needs. Architecture, because it almost explains all the functionality of the system. But the complexity of architectures varies from different parts to different elements. Therefore in following paper we tried to explain why to use ADM to modernize the system. To make it possible, one concept that is used most often is knowledge discovery metammodel. KDM specification By using both scenarios, we are eyeing to make it possible to modernize the existing system without increasing the complexity and maintaining the interoperability along with language and platform independance.

Index Terms— Legacy systems, ADM, KDM, Artifacts, Modernization, Architecture

1 INTRODUCTION

In todays world the need for making the systems more agile and flexible is increasing rapidly with the intent of interoperability, language independant and platform independant. There are existing systems which plays a vital role in organizations growth though they are old. The functioning, characteristics, working and complexity of these systems are totally different than the ones which are present today, still they are useful because of the capabilities they possess. One of the reason for their dissimilarity is the Architecture" they uses. These systems can be called as "Legacy Systems". A legacy System is an old method, technology, computer system, or application program. The legacy system may or may not remain in use even if it is no longer used, it may continue to impact the organization due to its historical role. The data that resides and processed through Legacy systems is archival type of data that can not be directly used and processed in the modern systems. So there is a need of mechanism which helps in converting or modifying the legacy systems into the new one. Typically there are many ways for doing this from which some are costly and some are very costly, because of this, small organizations are not capable of providing the fund to use these mechanisms. Through many inventions and discoveries the term Modernization" comes into the picture. When we convert any old thing into the new one we usually come across many problems in which some are not avoidable, to deal with this situation a solution must be available. If it is possible that existing systems or legacy systems can be converted into the modern systems with the help of some tools or mechanisms in a simple way then the cost and time of the users and organizations can get reduced. In this way we are able to change state of the old system into the new one, this concept can be called as the Modernization".

Today many organizations doing great work in making the system interoperable along with language and platform independance. This can be enhanced with the modernization of existing systems architectures. As we mention above architecture exists in many different forms, to use them we need a proper way of direction. Architecture can be defined in a number of ways with respect to their different geners. Software architecture have different features than the ones of hardware architecture. Architecture is helpful in deriving the important and useful characteristics of system/software along with its functionality. When we define any architecture we must take into account that the requirements should be fullfiled and there is a scope for modernizing that architecture. Still in many scenarios the definition of architecture is not defined in a proper way, so in different scenarios different definitions are used. In next section we try to define the accepted meaning of architecture and modernization.

1.1 Architecture

As the need is growing rapidly to maintain stability between legacy systems and the new systems, definitions of architecture has to be a precise and concrete one. In many cases the conflict arises because of amibiguity of required architecture due to which designers and programmers fell short to implement the needed software. If we try to define the architecture in a general way then it can be defined as "the process and product of planning, designing and construction". Like this in terms of IT it is the set of structures needed to explain about the software system, which comprise software elements, the relations between them, and the properties of both elements and relations. OR it is the conceptual model that defines the structure, behaviour, amd views of systems[4]. It de-

- *Amit Wasukar Assistant Prof. at JDIET, Yavatmal, M Tech in Computer Engineering, E-mail:amitwasukar@gmail.com*
- *Co-Author name is currently pursuing masters degree program in electric power engineering in University, Country, PH-01123456789. E-mail: author_name@mail.com*
  (This information is optional; change it according to your need.)

scribes the components and representation of the system organised in a proper way to describe the characteristics and the relationships among components which ultimately provides a blueprint for implementing and understanding the system developed.

In a broad way architecture of a software is termed as:

- The high level structure of a software system
- The way of creating such a high level structure
- Making documentation of such a high level structure

From literature point of view the understanding of software architecture can be gained as [4]

The software architecture of a program or computer system is the structure or structures of the system, which comprise software components, the externally visible properties of those components, and the relationships among them.

While this defnition provides the structural aspects of the software architecture, the defnition given by the IEEE 2000 standard emphasizes other aspects of software architecture, given as:

Architecture is the fundamental organization of a system embodied in its components, their relationships to each other and to the environment and the principles guiding its design and evolution.[4]

This defnition stresses that a system's software architecture is not only the model of the system at a certain point in time, but it also includes principles that guide its design and evolution. From above one thing is clear that architecture of a system is a vital thing to manage. Architectural decisions are among the first to be taken during system development and since they virtually affect every later stages of the development process, the impact of architectural mistakes results in high economical risk. One of the solution to avoid these mistakes is to evaluate the software architecture of a system against the problem statements and requirements specification.

1.2 Modernization

Today almost every organization and company runs system that have been implemented a long time ago along with the new age systems, and diffculties of oragani-zations lies within the simultaneous working of these systems. Adapting legacy software systems to new requirements often needs to make use of new technological advances. Mar-ket and business value of existing systems can only be preserved by transferring legacy systems into new technological surroundings. Migrating legacy systems, i.e. transfer-ring software systems to a new environment without changing the functionality, enables already proven applications to stay on rather just decaying it. This concept can be sometimes termed as"Modernization". The main emphasis is on the migration of old arhitecture to new architecture without changing its functionality and characteristics. Modernizing legacy systems services to new services enables both, the reuse of already established and proven software components and the integration with new services[1]. In order to gain most benefit from a migration, a comprehensive approach supporting the migration process and enabling the reuse of legacy code is required.

In a large view, use of this concept i.e. to modernize the legacy systems is done to overcome the below fears that are monopolising in the industries[9]

- Not more understands the old systems
- Replacement for legacy system or packages often fail
- IT is afraid to shut them off

We can get from above that understanding the working of legacy system is some what diffcult as compared to the new one because of their complex architecture and function-ality, so manpower should be well trained for handling this type of systems. This is one of the major reason for improper handling of legacy systems. If we try to exchange the legacy system with the another one, the probability can be anything for proper working i.e., in most of the cases directly exchanging the systems not worked, the reasons for this can be vary from anything to anything. Modernization is at the heart of many software organizations that seek to migrate from obsolete or aging languages and platforms to more modern environments. Modernization focused on transforming technical architectures that is, moving from one platform to another and / or from one language to another.

This task is diffcult if the old system is not giving any response for changing its state but the fact lies between finding the solution which will keep the system alive. Under certain conditions, if transforming the existing system to a different hardware or software environment is the most critical factor for making decision to reengineer, the migration of a software system is generally a reasonable approach to attaining the desired aims. In lots of cases the terms migration and modenization can be used as alternately because of their same meaning and functionality with some differences, hence major challenges of software migration are based on the fact that the migrated applications environment changes. From above understanding, we can get a clear idea that existing systems or legacy systems are diffcult to handle, so to get the maximum results from these systems there is a need of some middel mechanism. And if we are able to convert or modernize these systems, then it will be easy for organizations to save the time to learn the systems and cost of extra efforts envolved. Modernization does not come with single side, it has both sides going in its way. To add this along with advatages it come up with some disadvantage as well, if not properly handles. The best component to modernization process is the architecture of system, since it always helps and responsible to learn, manage, develop and change the systems basic functionality.

1.3 Why to use Modernization

It is known fact that the information systems are not static entities, but these are the dynamic entities means these systems changes over time as and when new technology emerges and the code in which they are written in also becomes an archive thing and leads to improper handling. So to tackle this situation, one solution comes in a way is to replace an old system with new one, but this approach is expensive and difficult to

get the return on investment. One of the main thing to take into consider that the business knowledge of legacy systems does not change when we replace it with new one.

Because of all this, the mentioned solution is not easily acceptable one. Therefore, there is a need of some evolutionary process which help us to make legacy systems workable with same business knowledge, same functioning, and same output. One solution that makes its way efficiently into market is the process of modernizing the legacy system. To tackle this situatuon Object Mabagement Group(OMG) proposed one standard related to architecture, the standard called as Architecture Driven Modernization". ADM basically works with artifacts of the legacy system which are the main elements, funcionalities of the system, arctifacts can be source code, user interfaces, databases, configuration files, etc. Therefore use of modernization becomes more useful than other techniques and in this report we are using this technique.

2. Architecture Driven Modernization

There is a large amount of highly functional and operational software representing enormous commercial value deployed in organizations around the globe. To be precise, existing systems are defined as an production enabled software, regardless of the platform it runs on, language it's written in, or length of time it has been in production. These entrenched software systems often resist evolution because their strategic value and ability to adapt has diminished or vanished through factors not exclusively related to its functionality. This leads to the systems inability to interoperate or dependance on undesired technologies or architecctures[10].

To support above problems we can use the concept of ADM. ADM can be used for the purpose of

1. Software improvement
2. Interoperability
3. Reuse
4. Modifcations
5. Restructuring
6. Migration
7. Translation
8. Integration
9. Service oriented architecture

Collectively, these activities can be defned as"Architecture-Driven Modernization" or"ADM". ADM is the process of understanding and evolving existing software assets. ADM restores the value of existing applications because it extracts and used it in further development to deliver new solutions that address changing business requirements[1]. In todays scenario ADM is very popular term among the organizations and companies, the facts which made it popular are just itemize above.

Modernization supports a series of scenarios like [9]:

- Application Quality Improvement[Standardization and Modularization]
- Source to Source Conversion
- Platform Migration
- System Consolidation
- Service Oriented Architecture Migration
- Model Driven Architecture Migration
- Application integration and data rearchitecting

Architecture Driven Modernization is driven by various vendor and user organizations seeking to share tool and analyst capture metadata from existing systems environment. Using ADM along with its different scenarios helps to make the systems more agile and effective. With this we can able to make systems/softwares platform and language independant along with interoperable. Interoperable systems can run on any given conditions. ADM is the best path to moving an Aging System to an Agile System when used with its best standards.

One of the standard or component of ADM which is used more oftenly in the modernization is Knowledge Discovery Meta-Model [KDM]. The KDM provides a common interchange format that allows interoperability between existing software analysis and moderniaztion tools, services and their respective models. This International Standard(ISO) defines a meta-model for representing information related to existing software, its elements, associations, and operational environments called "KDM"[2]. Through the use of a common meta model, each tool would be able to exchange common views across platforms and languages for the purpose of analyzing, standardizing and transforming existing systems. Architecture driven modernization can be used in making the legacy systems useful. Like by transforming old version of software into the new one with the help of some conversion tools. Historic data may not have been converted into the new system format and may exist within the new system with the use of a customized schema, or may exist only in a data warehouse. For a variety of reasons, a legacy system may continue to be used, sometimes well past its vendor supported lifetime, resulting in support and maintenance challenges. It may be that the system still provides for the users needs, even though newer technology or more efficient methods of performing a task are now available. A legacy system may include procedures or terminology which are no longer relevant in the current context, and may hinder or confuse understanding of the methods or technologies used.

Systems modernization has been providing benefits to organizations seeking to analyze software architectures in support of tactical systems initiatives such as software maintenance. Modernization has also delivered benefits to project teams seeking to migrate from an obsolete or aging languages and platforms to more modern environments. Modernization efforts are now reaching into more significant and impactful business domains, extending opportunities into the field of IT. Allowing modernization to reach its full potential requires a deeper understanding of its architectural impacts.

2.1 What ADM meant

Architecture Driven MOdernization(ADM) is used for the

modernization of the existing systems. ADM is the process of understanding and evolving existing software assets like its fuctionality, its components, mentioned requirements. Some of the existing systems are not capable to do the modernization, the reason for this can be anything since the complexity of architectures is a complex part to understand. Existing systems modernization is the process of understanding and evolving existing software assets to further one or more business or organizational objectives. ADM is used when existing IT practices fail to deliver against business objectives. Business objectives includes the competition among organizations, survival in the market. Modernization should be done from an analysis and design based perspective and not for source to source migrations, this is the main advantage associated with ADM. Modernization examines, exposes and facilitates the refactoring , redesign and redeployment of core application architectures with the intent of making cricital business requirement in a way that lowers risks , costs and delivery problems. The main use of architecture driven modernization comes in the form of

- Platform Independent
- Language Independent
- Interoperable

These are the important factors which enables organization to widely use ADM. Above mentioned characteristics specially helps the organizations to fullfill the needs of users and to survive in the market and to lead the market.
ADM has the capability to deliver[10] :

- High quality modernize system
- Fraction the time and cost of the past methods
- Handles combination of source and target languages
- Applicable to
    - Commercial
    - Governmental
    - Military

Because of the conflicting or confusing designs and laborious manual coding methods the business agility of an organization degrades which leads to the loosing shares in the market.

2.2 Benefits Of using ADM

ADM is vastly used term today because of its functionality, so definitely it possesses some of the major concepts or functions and by using these concepts organizations mark their way towards some improvemental standard.
The advantages we can get by using the concept of ADM are[10]

- Enabling business agility by creating software agility
- Improving ROI in existing software
    - Improving productivity of software development
    - Reducing maintenance effort and cost
- Interoperability
- Adoption of new technologies, Practices and paradigms
- Modernization is done on the basis of prior experiences and best practices
- Standardization in modernization can lead to
    - Integration and interoperability between different vendors and tools by creating open framework
    - Encourage collaboration among complementary vendors
    - End users will get better operational cost
- Reduces the time, cost and risk of software transformations

2.3 IT Architecture Modernization

A strong cross section of software tool vendors and service providers has emerged to enable the modernization of existing systems. Unfortunately, users and vendors have been working in isolation, often reinventing the wheel. Modernization results and processes build on prior experiences and best practices. ADM specific ations and corresponding tools will enable projects with large and/or aging software to become more agile. There is a need for modernization projects to enable integration and interoperability among solutions from multiple vendors. Modernization will increase interoperability between different tools and manual processes by creating an open framework. This will enable a new generation of solutions to benefit the whole industry and encourage collaboration among complementary vendors[9]. ADM standards will allow users to begin modernization projects knowing that there is interoperability among different tool vendors. In other words, standardization will ensure that end users are investing not just in individual tools but rather in a coordinated modernization strategy.

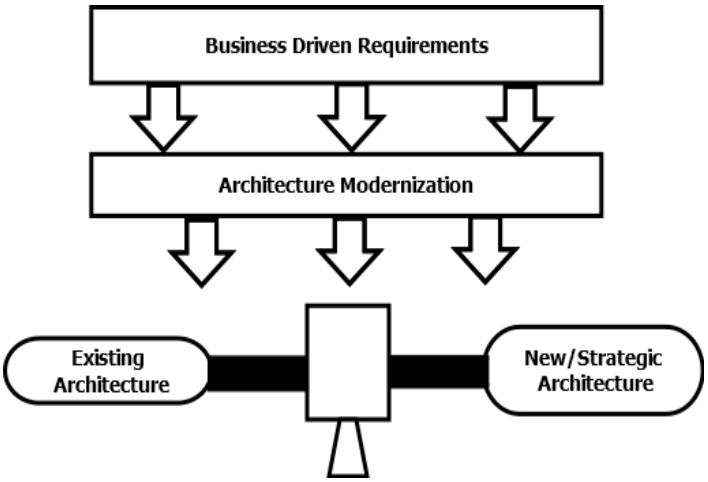

Fig. 2.1 Architecture Bridging

There are many long term strategic reasons to work towards modernization of existing systems. In the past, the forward engineering industry has largely ignored existing software assets, while reverse engineering was in turn ignoring new forward engineering methods. Over time, this has created a

significant gap between modernization tools and methodology support for projects involving maintenance and evolution of existing systems. ADM will help break the barriers between forward engineering tools and reverse engineering tools and help organizations with existing software assets realize the benefits of modern forward engineering technologies, such as the Unified Modeling Language (UML) and the Model-Driven Architecture(MDA).

The diagram shown in fig 2.1 describes the bridging of two architectures. The architectures involved are existing/legacy architecture and modernize architecture. As we know modernization bridges the gap between old and new systems, the fact that lies is between the working of these systems. Firstly one has to know the functional and business driven requirements needed for the modernization of the systems architecture. The requirements are varying from one type to another depending upon the user needs and organizational progressing. After knowing it we can apply our architecture driven modernization concept to it and after all of this we clearly see the difference between existing system and modernize system.

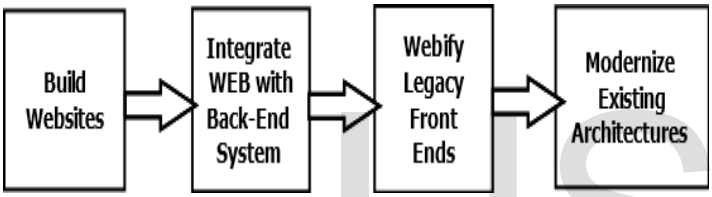

Fig. 2.2 Architecture Evolution

The diagram 2.2 depicts the IT architecture evolution. In todays world lot of things are available and can be carried out through the web services. So there is a need to make sure that things has been properly deployed on web.As shown in diagram first websites are build and as per services provided proper back end is used to integrate it to the web. It is always the matter of concern that the Front end of the existing system/legacy systems are not compatible with new systems. Therefore we have to change the legacy system by means of proper web designs and after webifying the legacy system we can modernize the existing system in the way we want.

3. Knowledge Discovery Meta Model

The Knowledge Discovery Metamodel(KDM) is a standard to support Architecture Driven Modernization (ADM). This standard provides a specification at a level of abstraction higher than source code in order to specify a language independent representation of programs. The KDM currently supports a wide collection of programming languages and is an intermediate representation of software artifacts and its operational environment. The KDM specification can be used as a language and platform independent representation for analyzing software systems[6]. This allows the incremental analysis of a software system, where each tool reads and analyzes the KDM representation, extracts precise knowledge and if required adds more knowledge to it.

3.1 Layers in KDM

The metamodel of the KDM standard defnes four layers[6]:

- Infrastructure Layer
- Program Elements Layer
- Runtime Resources Layer
- Abstractions Layer

Each abstraction layer of the metamodel is based on the previous layer. Furthermore, each layer is organized into packages that define a set of metamodel elements whose purpose is to represent a specific independent concern of knowledge related to legacy systems.

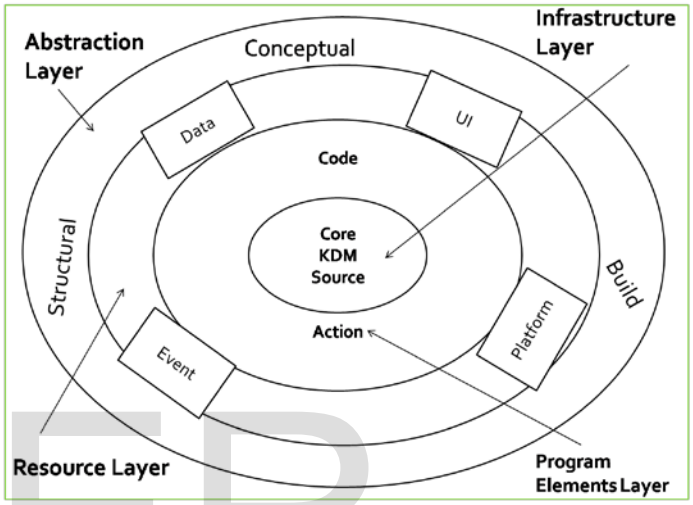

Fig. 3.1 KDM Layers

Infrastructure Layer

It is the lowest abstraction level, which consists of a small set of common metamodel elements (such as entity and relationship) used through entire KDM levels. This layer consists of three packages: Core, KDM and Source

Program Elements Layer

It represents the code elements and their associations. It consists of a broad set of metamodel elements common between different programming languages to provide a language-independent representation. There are two packages in this layer: Code and Action.

Runtime Resource Layer

It represents higher level knowledge (such as operational environment) about existing software systems. This kind of knowledge cannot be extracted from the syntax at code level but rather from the runtime incremental analysis of the system. There are four packages in this layer: Data, Event, UI and Platform.

Abstractions Layer

It defines a set of metamodel elements for representing domain and business-specific overview of the system. Extracting this kind of knowledge involves input from experts and analysts. Conceptual, Structure and Build are the three packages present in this layer.

The key design characteristics of KDM are :

- KDM is an Entity Relational Model
- KDM can be extended to capture language specific , application specific and implementation specific entities and relationships.
- KDM models are composable

4. Applications of ADM through KDM

If we consider the overall view then the need for using legacy systems is increasing rapidly, to cop-up with this scenario, organizations have to use the modernization of legacy systems. The reason behind it, is the functionality of ADM which changes the game of organizations to persist in the market place. The number of applications for using ADM through KDM is:

- It is used for making the systems platform and language independent.
- It also makes system interoperable.
- Because of the layered organization the understanding of architecture processing and software artifacts becomes easy.
- The KDM standard makes it possible to manage the legacy knowledge recovered from a legacy system throughout all the stages in an ADM based process.
- The knowledge recovered from legacy systems and represented in KDM models can be shared by different modernization tools.

5. Conclusion

If we are able to make the old systems usable with some minor changes to its architecture, then lots of things gets easy for organization and user also. Because to handle old systems the project team needs to be super expertise, to avoid this we can use a concept, to transform the old system. To achieve this we use the term "Architecture Driven Modernization" along with its specification called "knowledge discovery metamodel". We have tried to understand the basic designing of architecture and tried to transform it into the modernize one. ADM supports many functions like language and platform independane along with interoperability between the systems. In further proceeding, we are trying to explore the working and usability of KDM for modernizing the legacy systems.

REFERENCES

[1] W. Ulrich.(2004), ``A status on OMG Architecture-Driven Modernization task force," \textit{In Proceedings EDOC Workshop on Model-Driven Evolution of Legacy Systems (MELS)}, Monterey, USA [IEEE Computer Society Digital Library].

[2] Remco C. de Boer, ``Architectural Knowledge Discovery Why and How?", Department of Computer Science, Vrije Universiteit, Amsterdam, the Netherlands, remco@few.vu.nl

[3] Ga¨etan Deltombe Netfective Technology Software 32,avenue L´eonard de Vinci 33600 – Pessac, France, g.deltombe@netfective.com, ``Bridging KDM and ASTM for Model-Driven Software Modernization"

[4] Banani Roy and T.C. Nicholas Graham, ``Methods for Evaluating Software Architecture: A Survey", \textit{Technical Report No. 2008-545}, School of Computing Queen's University at Kingston Ontario, Canada April 14, 2008

[5] Information technology - Architecture-Driven Modernization, OMG. Architecture-Driven Modernization (ADM) / Knowledge Discovery Meta-model (KDM) 1.2 Specification. on (ADM): Knowledge Discovery Meta-Model(KDM)

[6] ``OMG. Architecture-Driven Modernization (ADM) / Knowledge Discovery Meta-model (KDM) 1.2 Specification", \textit{Technical Report formal /2010-06-03,}, http://www.omg.org/spec/KDM/1.2/, 2010

[7] Architecture Driven Modernization (ADM), http://adm.omg.org

[8] Avi Yaeli, Netta Aizenbud, Jonathan Bnayahu, Nurit Dor, Alex Akilov , Sara Porat, ``Legacy MOdernization to SOA using Compass/VB - Case Study", \textit{IBM Research Labs in Haifa Jenny Choy, IGS}

[9] William M. Ulrich, ``Architecture Driven Modernization 101: Concepts, Strategies \& Justification", \textit{Tactical Strategy Group, Inc.}, www.systemtransformation.com

[10] ``Why do we need standards for the modernization of existing systems?", \textit{OMG ADM Task Force