# Legacy Transformation - Ways to Reactto Market Forces

**Debiprasad Mukherjee**

**Lead Business Consultant**

**Ericsson India Global Services Pvt. Ltd.**

## Abstract

*Different strategies clients are implementing for IT systems to meet market demand, with one extreme being replacing the whole system to in-between approach of replacing or fixing one application at a time to other extreme of modernizing with the old technology or do no system modernization.Market forces are demanding every small & large company in every corner of the world to respond to changes in a manner it has never done before. Let it be the Consumerism, Digitization, Social-Mobile-Analytics-Cloud (SMAC) or Productization, the IT systems must be agile enough to respond to market demand in a better–faster–cheaper ways. However, not all the current day legacy applications, built with technology & architecture which dates back to decades can support this continuously accelerating pace of change. This article describes the five different ways clients are approaching legacy modernization to stay in the race (based on the implementation exposure on legacy migration initiatives across geographies for different industries).*

**Keywords**

*Consumerism, Compliance, Consolidation, Digitization, Legacy Migration, Transformation*

## Legacy Modernization

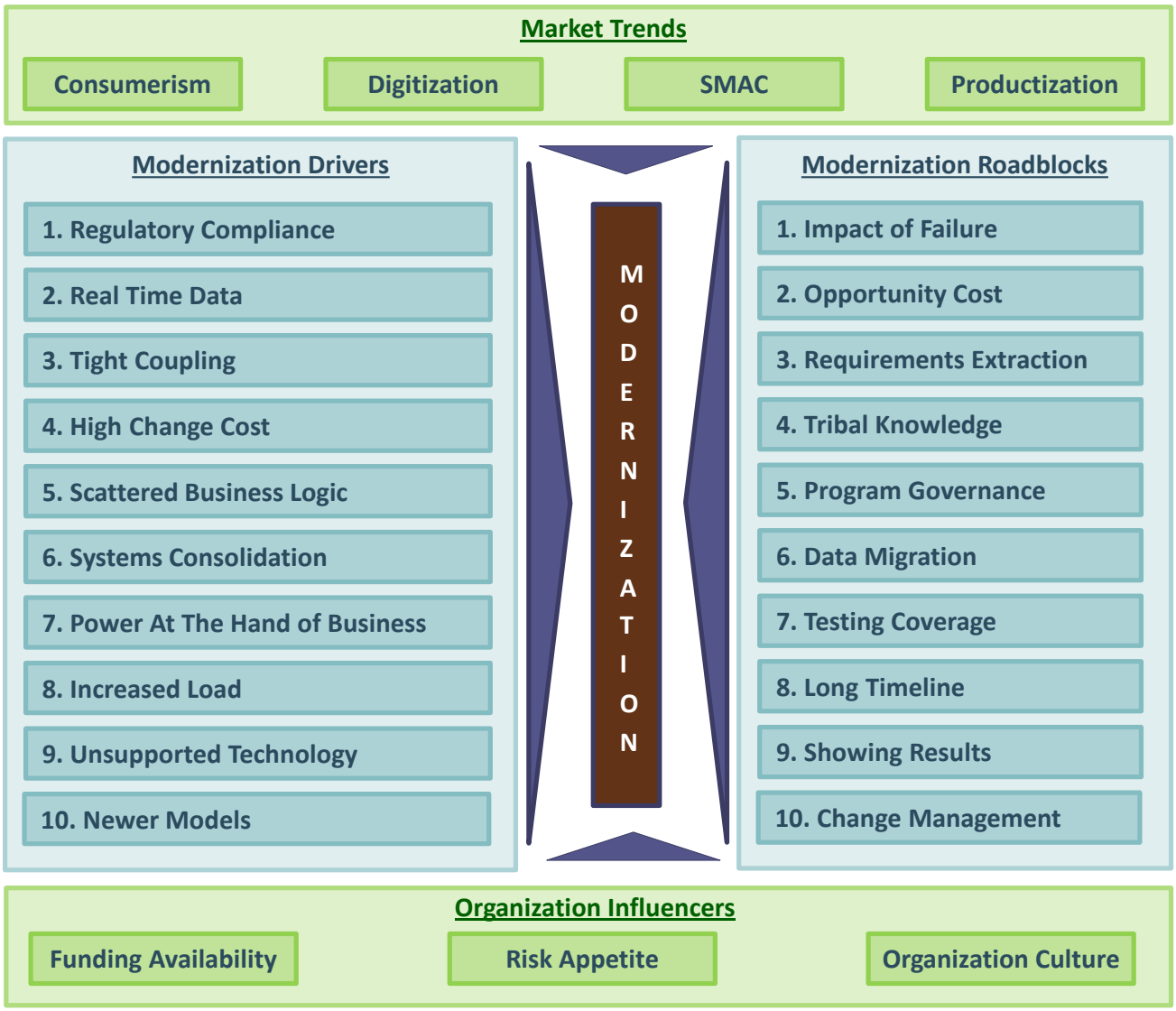


Fig1- Modernization Framework

### Modernization Drivers

Following are the top 10 reasons which we have seen to trigger legacy modernization drives

1. **Regulatory Compliance**: Regulatory changes are very frequent and come with tight timelines to respond. Many systems cannot handle so much churning in a short time and any miss in compliance deadline may lead to penalty & sanctions.
2. **Real Time Data**: Earlier systems were heavily batch processing based where data would typically be updated daily. However, with increased use of internet and mobile devices, users now want real time data much more than ever before.

3. **Tight Coupling**: Systems & modules were built entirely based on the given set of applications in mind and changing one component would require changing the whole universe. This is opposite of modern day Service Oriented Architecture (SOA) paradigm.
4. **High Change Cost:** Legacy systems used common code modules whose changes require recompilation & retesting of multiple linked modules increasing both development & testing costs. With increase in change frequency, it increases the priority of cost containment.
5. **Scattered Business Logic**: Business rules are often written all over the place and any small change require a lot of impact analysis and runs the risk of missing changes with serious business impacts. The current trend is to use centralize rules engine to address this challenge.
6. **Systems Consolidation**: Often clients have multiple systems for single functionality due to mergers & acquisitions or lack of coordination in units (e.g. states, LOBs). Any change needs to be repeated in each of these systems increasing the cost. System consolidation is done to remove this duplicity.
7. **Power At The Hand of Business**: Business wants to make frequent, small changes to production with no/low IT involvement. It is not possible in old systems as it requires programming knowledge. BRMS tools allow rules delegation & Excel type rules editing that business can do on their own.
8. **Increased Load**: With increase of IT usage & user volume, the load on IT systems are increasing fast. Old architecture may test the limits of scalability & performance leading to user dissatisfaction & availability challenges. New infrastructure is often the best solution.
9. **Unsupported Technology**: Every hardware & software has a limit on lifespan & support from the vendor who built it. Legacy applications built decades back runs the risk of expiry of vendor support which makes the systems vulnerable to run any mission critical application.
10. **Newer Models**: With advent of Cloud and other hosting models, clients are exploring just in time capacity models to handle the seasonal business volume. Older architectures are not compatible with these newer platforms and often had to pave the way for new ways.

**Modernization Roadblocks**

However, coming out of these long running systems is a monumental task and can lead to failures if not planned & executed properly. Following are Top 10 challenges to overcome

1. **Impact of Failure**: Typically the legacy applications support the client's core functions which are mission critical. Any impact to the application functionality or performance of availability can lead to serious business impact and loss of user confidence on the program.
2. **Opportunity Cost**: Large transformations require huge investments which are often difficult justify through concrete ROI. Funding one such program will deprive budget from many other equally important opportunities stalling growth in other areas.
3. **Requirements Extraction**: Generally, requirements are not available for the legacy applications and requirements & rules extraction through reverse engineering is an

extremely challenging job. Equally difficult is to validate the correctness & completeness of these requirements.

4. **Tribal Knowledge**: Often, applications are built in a cryptic way which only those involved in design can clarify. With time & attrition, this tribal knowledge gets lost. During replatform, understanding & clarifying this functionality becomes a big challenge.
5. **Program Governance:** A transformation program involves several teams from different areas and multiple stakeholders to address. Making sure all of them work as one team towards one goal thorough program management, planning, tracking & communication are keys to success.
6. **Data Migration**: Understanding, mapping & migrating historical data to new systems is a common failure point. Handling unexpected or incorrect data and data cleansing is a high risk task. Any issue in the system of record will make the entire application unstable.
7. **Testing Coverage**: End to end systems testing to validate functionality, integration & data correctness is a colossal task. It requires through test planning, in depth domain knowledge and flawless test execution & validation. Failure in any of these can lead to quality issues.
8. **Long Timeline**: Transformation programs span over multiple years requiring continuous focus for a long time. It must support business as usual and handle on the flight changes. It should also weather all tactical/ad hoc changes in old systems and potential leadership change.
9. **Showing Results**: Initial phases like requirements & design can run for months and sponsors may not see any tangible results unless first deployment happens. It can lead to sense of discouragement in business and should be handled through incremental visible results.
10. **Change Management**: Organization change management is a deal breaker for transformations. Moving end users from comfort zone & old habits to new systems, especially new UI is not easy. IT folks should be reskilled and company culture should be modernized to match the modernized systems.

**Organization Influencers**

Different clients respond to these risks and plans the modernization in differs ways based on following organization characteristics

- **Funding Availability**: End to end transformation program requires large scale funding over multiple years. On the other hand program fund sourced by specific initiatives (e.g. compliance) lead to modernization of impacted applications only.
- **Risk Appetite**: Companies with higher risk tolerance try to achieve larger returns through larger scope and adoption of newer technology and becomes a leader. Conservative companies typically go for small scale changes over a long period of time and remain follower.
- **Organization Culture**: An organization with an atmosphere of innovation encourages large scale changes more than the company which tends to remain low cost provider

through low budget for technology. Often leadership's inclination towards a technology determines the direction.

**Modernization Approaches**

Following are the 5 different ways we have observed clients approach the modernization based on above characteristics.

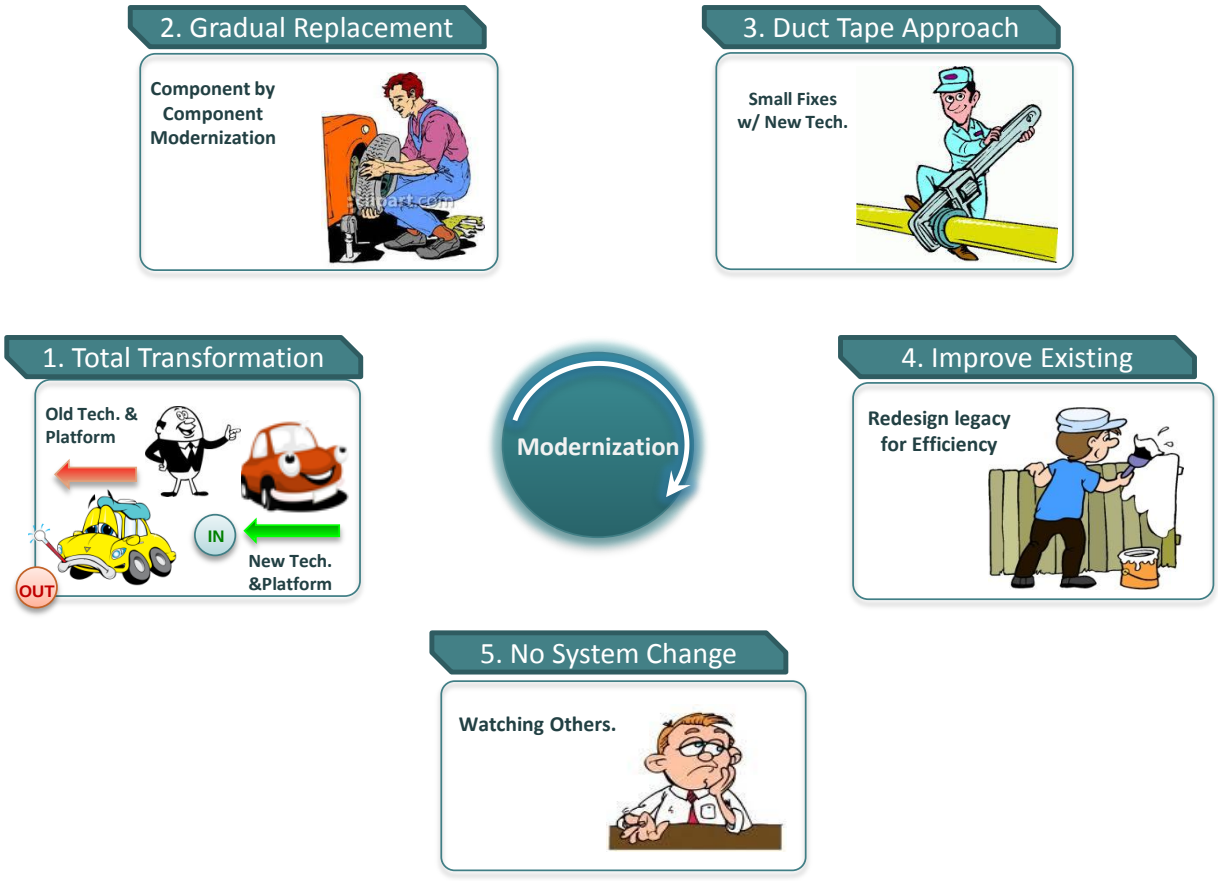


Fig2- Modernization Approach

## 5-Ways of Modernization

Following is a high level overview of the 5 modernization approaches

### 1. TOTAL TRANSFORMATION

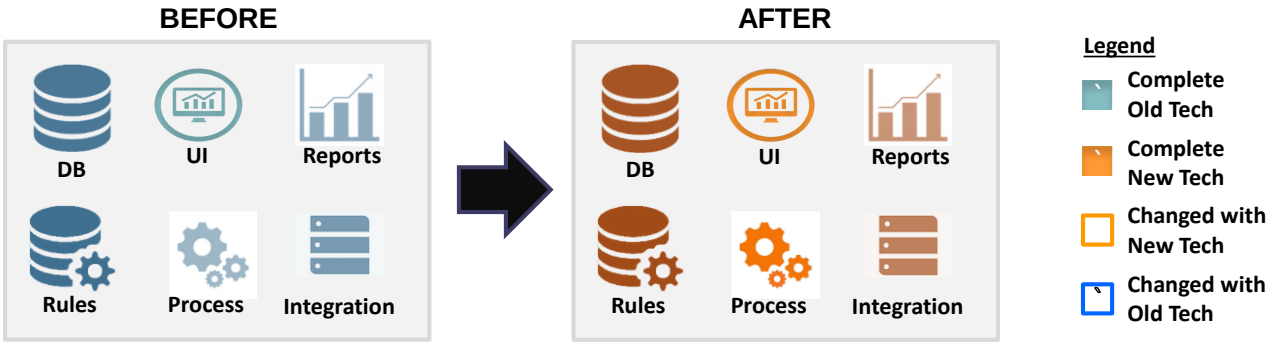


Fig3-Total Transformation

| Strategy | |
|---|---|
| **Description** | ▪ An entire system is rebuilt using new technology. Old system (s) is sunset and the new system takes it place to support that (& potentially additional) functionality.<br>▪ The new system can be built from the scratch using standard platforms (e.g. J2EE) or more commonly built with a third party package as foundation layer to get a jump start.<br>▪ The change is rolled over to production either in a single go live or preferably in phases |
| **Example** | ▪ End to end Healthcare Claims system replacement from Mainframe based home grown application to new Digital platform based on BPM package & frameworks |
| **Applicable Situations** | ▪ Clients are running on an age old system which cannot be continued any further (e.g. platform is getting out of support) and need to move to a totally new systems.<br>▪ Current system is built on a product which has high license cost and an alternate, potentially open source, product is used to rebuild the system to bring the IT operational cost down.<br>▪ Clients have gone through mergers & acquisition and none of the systems can support the combined entity which necessitates building a new application with unified functionality<br>▪ Clients are pursuing aggressive business goals to lead their market segment and system capabilities must scale up to support that vision.<br>▪ New IT leadership is eager to bring a paradigm shift in IT capability & operation and shared new vision & mission statements to modernize the systems using new set of technologies. |
| **Benefits** | ▪ This approach gives the highest return & highest edge over competitors. It allows business to build the system exactly the way they want and enable them to respond to market forces.<br>▪ Since the latest technology stacks are chosen with futuristic views, the system technology stack & architecture will be ahead of the curve for several years in future.<br>▪ Rebuilding the entire application from ground up gives the maximum flexibility in |

| | |
|---|---|
| | architecture & design and the system doesn't need to be constrained by old components.<br>▪ All operational measures, SLAs/KPIs, business effectiveness & efficiency typically get facelift in new systems.<br>▪ This gives an excellent opportunity to redefine/improve the business processes leveraging new systems functions not available earlier. |
| **Success Factors** | ▪ Since clients are making a huge commitment to a new technology, proper due diligence / assessment must be done in selecting the technology.<br>▪ Since this is typically a multiyear program, clear roadmap with key milestones (e.g. go live dates) must be defined upfront. This can be done through an assessment phase<br>▪ It is better to roll out the change in phases to minimize the volume & impact of change compared to single big bang go-live.<br>▪ Stakeholder management is key as a change of this magnitude must be communicated properly and accepted by all<br>▪ End to end functionality & integration testing must be done thoroughly. Production parallel execution & comparison of old & new systems can identify any misses in testing.<br>▪ Understanding the data structure, quality & remediation is key if data migration is involved.<br>▪ This opportunity should be leveraged to Capture the non-functional requirements (e.g. availability, performance) in details and perform load & performance testing to validate that.<br>▪ Since the application is most likely rebuilt using a new technology, the IT department must learn the new technology stack to support & maintain it going forward. |
| **Risks** | ▪ This is the most risky, yet most rewarding approach.<br>▪ Since the entire application is being rebuilt, the entire functionality is at stake and any major quality issue can impact the business operations drastically.<br>▪ These are large scale programs and if not tracked properly can run into serious delivery issues. Strong program management & governance practice must be followed.<br>▪ Day to day job of end users from business & operations are impacted heavily by this change and difficulty to accept the change & pockets of resistance can derail the program.<br>▪ Since the timeline of the programs is typically large, business sponsor may lose motivation if no incremental business benefits are shown at regular periods. |
| **Verdict** | ▪ It is the best option if it is budgeted, planned & executed properly as it takes the entire system several notches ahead and gives competitive advantage over peers for several years. But, the rate of program failure or midway stop/hold is also highest in this approach. Hence only the organizations with required IT capability & maturity should go for this route. |

## 2. GRADUAL REPLACEMENT

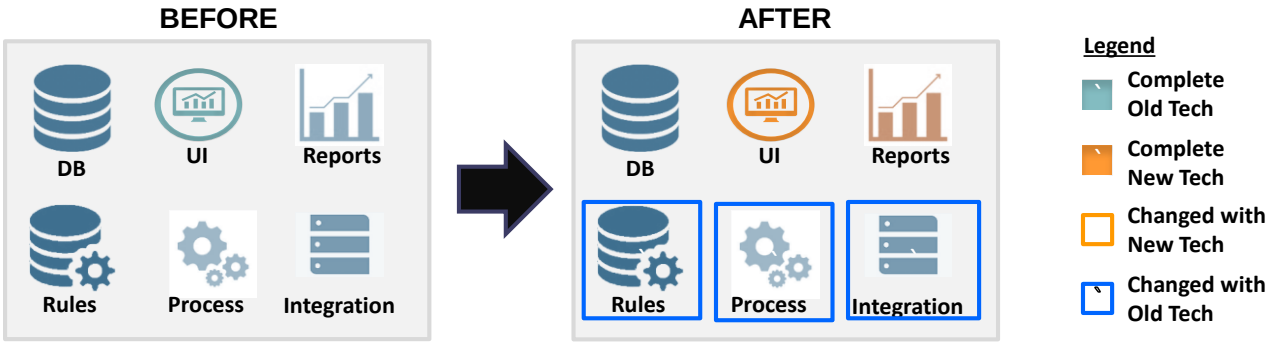


Fig4- Gradual Replacement

| Strategy | |
|---|---|
| **Description** | ▪ When a particular component/functional block needs modernization, it is selectively replaced with a new technology while rest of the system remain on old technology<br>▪ The newly built application is moved to production as a separate entity and delivers a unique set of functionality to customer.<br>▪ The new application is integrated with the remainder of the systems following a SOA model.<br>▪ Over the time more & more such components/ functional blocks are replaced with new apps and gradually the entire application is replaced with a set of connected applications. |
| **Example** | ▪ To support a new regulation & real time data needs by end user, Claims Accumulator functionality (i.e. to calculate deductible etc.) is taken out of an end to end Legacy claims systems and built as a separate application using a BPMS & BRMS platform. |
| **Applicable Situations** | ▪ Clients are planning to replace the entire systems but due to controlled release of budget and to keep the impact & risk low, they are modernizing one component at a time.<br>▪ A component of an application / systems is not being able to support the market demand and clients selectively modernize this component to meet the need.<br>▪ Business rules are scattered in legacy systems and clients took the initiative to externalize rules using BRMS platforms. It also enables business to update the rules themselves.<br>▪ Clients want to decouple the systems and improve the integration through implementation of Enterprise Service Bus (ESB) and loose coupling.<br>▪ Clients changing the existing systems from batch to real-time/online and data update/ synchronization process is changed to support that<br>▪ Data maintained in the systems is required other systems and a service wrapper (e.g. web services) on top of the existing systems is built to provide |

| | |
|---|---|
| | the data.<br>▪ Legacy systems maintained data in flat files or outdated database systems which are changed to modern day leading RDBMS systems.<br>▪ Older systems was using legacy dumb terminals which are modernized using new Internet browser based UI which can also be extended to mobile devices and tablets.<br>▪ Existing systems using a technology where the license cost is very high and clients are replacing that component with an open source technology. |
| **Benefits** | ▪ This is a low risk way to get the entire systems transformed by moving one piece at a time.<br>▪ This requires much lesser one time budget approval compared to total transformation.<br>▪ Since one functionality is migrated at a time, the business impact & cost of failure are lower.<br>▪ Since work volume is less than total transform, it consumes less management bandwidth.<br>▪ This delivers results quicker compared to total transformation as end users get a complete functionality working in production. This encourages the business sponsor for more funding.<br>▪ This helps clients to break one large systems into a SOA based componentized systems which can be changed/replaced in future with low/no impact to rest of the applications.<br>▪ SLA/KPI and other operational measures can now be tracked at more granular levels<br>▪ Since each component is built separately, this allows clients to choose different technologies or newer versions for later components. |
| **Success Factors** | ▪ It is a standard replatform program where a new app will be built based on requirements from an existing app and all relevant program management practice to be followed.<br>▪ Since more applications like this will be built in future to replatform other functionalities, team must explore ways of building reusable components or other artifacts.<br>▪ There must a clear technology direction so that all new application use same set of technologies and follow consistent architecture & design principles.<br>▪ There should be a governing body to ensure all these new applications are tied to common goal & roadmap and typically steering committee or CoE is established to enforce that.<br>▪ Continuity of key resources long term to support multiple new application development.<br>▪ Smooth communication among different concurrent application teams is key for consistency.<br>▪ Nonfunctional requirements must be captured for all these applications and load & performance testing should be planned & executed. |

| | |
|---|---|
| **Risks** | ▪ Risk of amalgamation issues among different applications if respective teams are not connected & governed closely.<br>▪ This can lead to a set of disjointed applications which serves its functionality but doesn't function well as a unit.<br>▪ In absence of close monitoring and centralized technology gate keeping, each application can land up using different technologies to meet similar functionality.<br>▪ If multiple components are replatformed simultaneously, the concurrent development can lead to versioning issues of common/ reusable components and conflict in design decisions.<br>▪ These are medium to large size projects and are associated with typical delivery challenges like tight timeline, requirements completeness, testing coverage, code quality etc. |
| **Verdict** | ▪ This is one of the best approaches to follow as it slowly brings in new technology while keeping the change impact & risk to low. Success rate is very high in this approach. |

### 3. DUCT TAPE APPROACH

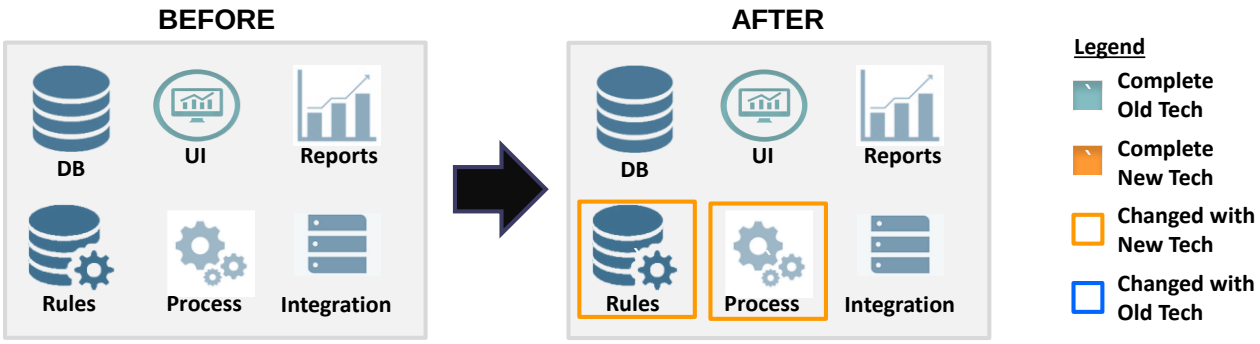


Fig5- Duct Tape Approach

| **Strategy** | |
|---|---|
| **Description** | ▪ Localized, small scale changes are done to address specific issues in the application.<br>▪ The application core architecture and technology remains the same and few components are modernized or rebuilt to deliver the results.<br>▪ Potentially this new component is rebuilt using a new technology<br>▪ One popular approach is to build a new application or component which will be bolted to the main application to bridge the gap in the functionality. |
| **Example** | ▪ Increase Claims Auto-adjudication ratio (i.e. % claims processed automatically with no manual intervention) by automatically correcting the Pended claims (i.e. flagged for manual correction) from a legacy Claims systems using a separate rules engine driven application. |

| | |
|---|---|
| **Applicable Situations** | ▪ Clients plans to continue with the legacy application but wants to fix the existing issues and augment missing functionality through introduction of a new technology.<br>▪ This focuses on current problems that must be fixed today and cannot wait for large scale, long duration projects. Improving KPI/Metrics through easy fixes is most common example.<br>▪ Clients have identified a problem in the middle of the year which was not considered in annual planning and hence no large budget is available for comprehensive solution.<br>▪ Clients are looking for a Stop-Gap solution to fix current issues while they plan for modernizing or replatforming or obsoleting the application in future<br>▪ Clients have undertaken large programs in another area which doesn't leave much bandwidth and risk appetite for rest, yet have to meet the demands of market. |
| **Benefits** | ▪ Small scale changes offering big returns as it addresses the core issues that can be fixed with simple solutions.<br>▪ Doesn't require huge investment and can be supported through Ad Hoc budgets (compared to other approaches which are typically funded through annual planning in previous year).<br>▪ ROI is always concrete and substantial. Increases the confidence of the project sponsor.<br>▪ Offers quick wins as results are often delivered in months. Increases end user morale.<br>▪ Less risky approach and probability & cost of failure are typically low.<br>▪ Doesn't require as much management attention as larger transformations do. |
| **Success Factors** | ▪ There must a strong Technical governance structure to ensure this easy-to-follow approach is not overused and alternate approaches should be given due consideration.<br>▪ A tactics works best when coupled well with strategy. Careful attention should be given for future roadmap/strategy of the application to ensure this tactical change is not a sunk cost.<br>▪ If the application requires more changes / replatform in future, it may be cost effective and better design to replatform or rebuild the whole application.<br>▪ No matter how small the change is all project management processes must be followed properly to ensure it is planned & tracked to success. |
| **Risks** | ▪ Too much of this approach in too many places in an application can lead to patchwork type changes and can lead to overall bad design.<br>▪ If multiple duct tape changes are done in an application, over the time total cost incurred may be substantial and can be higher than what it would require to replatform the same.<br>▪ A change done with no due diligence can lead to Throw-Away work as the Stop Gap solution may not keep pace with Changing Business Needs over time. |

| | |
|---|---|
| | ▪ Since the cost & impact of these changes are not too big, often these do not show up in management Radar and may miss the typical review process all other projects go through. |
| **Verdict** | ▪ It is one of the most common & successful approach in practice, as it gives quick wins with comparatively low investment & low risk. However, it still remains as a tactical solution and works best in combination with a good overall long term strategy. |

## 4. IMPROVE EXISTING

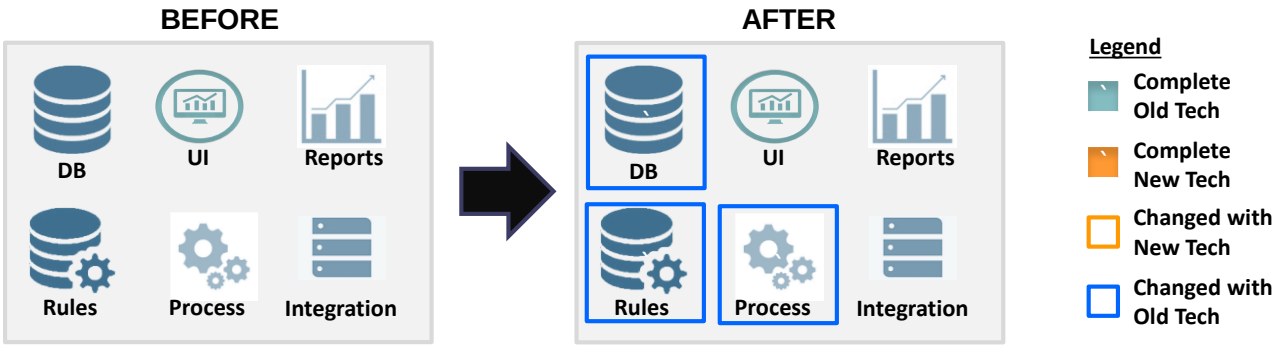


Fig6- Improve Existing

| **Strategy** | |
|---|---|
| **Description** | ▪ The existing system is modernized to offer better results through improved design.<br>▪ Typically the core technology stack remains the same or may introduce few minor additions. |
| **Example** | ▪ Improve application code maintenance by consolidating scattered & repeated business rules into common components and removing dead / redundant code. |
| **Applicable Situations** | ▪ Leadership has strong association with legacy app and wants it to continue while finding ways to improve its performance. Future leadership change will change direction.<br>▪ Client's IT folks are highly skilled in old technology and they keep finding ways to deliver continuous improvement. Gradually they will reskill to newer platforms and bring new stack.<br>▪ Technology foundation is modern (e.g. Java/J2EE) though the application may be old. The recent advances of the technology (e.g. frameworks) can be leveraged for improvements.<br>▪ Core platform has launched a new version which brings in new features and the application is migrated to this new version and synergy is used for enhancements (i.e. low hanging fruits)<br>▪ System is not performing well not due to the technology limitation but due to |

| | |
|---|---|
| | poor design/ architecture which can be improved through selective component redesign |
| **Benefits** | ▪ This extends the lifespan of the application by few more years while attending immediate burning issues. Clients can use this time to plan for future changes.<br>▪ Clients do not need to make a decision on future technology immediately as the changes are done using old stack.<br>▪ Client's existing legacy support team can anchor this change and no immediate need to reskill / learn a different technology.<br>▪ It doesn't require to procure any new technology and can avoid any new license cost |
| **Success Factors** | ▪ Modernizing legacy technology based systems is a matured practice going on for decades and clients should consult service providers experienced in this matter.<br>▪ Clients should leverage this opportunity to make the application componentized & SOA aligned so that individual components can be easily replaced in future if required.<br>▪ Proper documentation of business requirements and required testing needs should be done for better understanding of the application and jump start for any future replatform.<br>▪ Any major investment in legacy systems should be carefully reviewed to ensure this is in line with the client's technology vision & roadmap |
| **Risks** | ▪ If the system lifespan is limited, any large investment on the application can be a sunk cost.<br>▪ This might open up the floodgates for doing fresh development using an expiring technology.<br>▪ Scarcity of experienced resources in an old technology can force clients to deliver the projects using inexperienced / trained resources which can lead to quality issues. |
| **Verdict** | ▪ Every system comes with an expiry date which can be months or years or decades away. If a systems was built long past, it is unlikely that it will continue for indefinite period. An initiative like this can inject few years of life which can give just enough time for clients to get ready for embracing new technologies. |

## 5. NO SYSTEM CHANGE

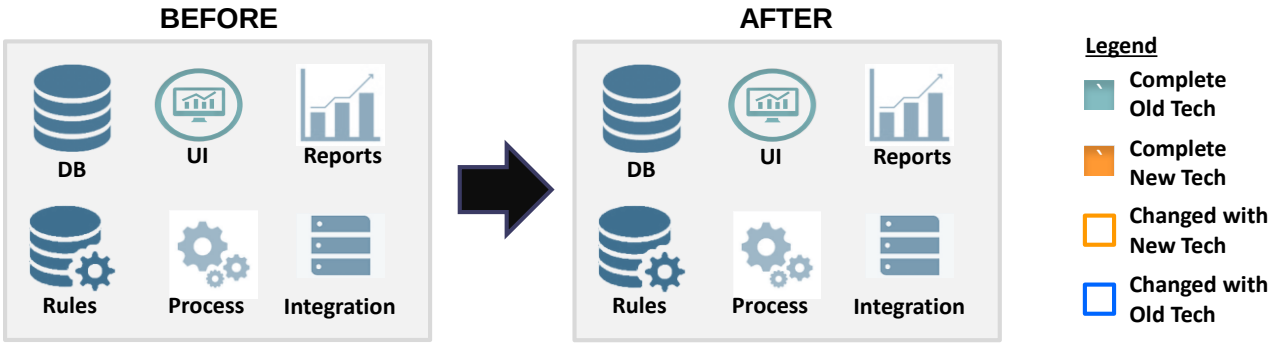


Fig7- No System Change

| Strategy | No Systems Change |
|---|---|
| **Description** | ▪ Clients are taking wait & watch strategy and not going for any major modernization drive<br>▪ Clients may have done the ROI calculation to infer current cost of inefficiency is not strong enough to compensate for the cost of modernization initiative<br>▪ No changes to systems other than usual production support & maintenance and minor enhancements |
| **Example** | ▪ Clients analyzed the ROI of claims systems modernization is not too high and decided to wait for more compelling business case in future. |
| **Applicable Situations** | ▪ Clients have modernized in the past or it is a new company with new systems. They believe the current system is good enough to support the market demands for immediate future.<br>▪ Clients are analyzing which is the best course of action for them and working on the scoping & roadmap definition. From outside it may look as if they are not doing anything.<br>▪ There are changes in leadership or company structure and the company is focused on absorbing the change. They will go for modernization once everything is stabilized.<br>▪ Leadership is indecisive and not being able to make their mind or come to consensus on which approach to take. Often it requires a top down resolution to finalize the strategy.<br>▪ Leadership believes they do not need technology advancements to support growth. While it may work in a short run but over time it will lead to serious competitor disadvantage.<br>▪ Lack of funding availability and it better be a temporary problem. It can become catch 22 situation as modernization funding requires profit which requires strong systems support. |
| **Benefits** | ▪ No need of investment and client can make use of the available money for other purposes.<br>▪ In absence of large IT projects, IT department has available bandwidth which |

| | |
|---|---|
| | can be used for future planning & modernization<br>▪ Since no technology decision is made, clients retain the flexibility to close any new platform |
| **Success Factors** | ▪ While clients may not be doing any systems changes but they can go for business process optimization to derive some benefits & efficiency.<br>▪ Client must continue to observe how the industry is responding to market forces and approaching modernization so that they are ready before starting their own drive. |
| **Risks** | ▪ Client's will operate with huge competitive disadvantage and runs the risk of falling behind the peers in quickly responding to market forces<br>▪ Due to lack of efficiency & optimization brought by modern systems, clients business processes more likely to cost more and lack the required flexibility<br>▪ This can lead to losing both market share and mind share |
| **Verdict** | ▪ Unless clients have a definite plan to make sure the systems are agile enough to support the growth, following this approach for too long can be dangerous to the future of the company. |

**Approach Comparison**

Following is a high level comparison of the 5 modernization approaches

| | 1. Total Transformation | 2. Gradual Replacement | 3. Duct Tape Approach | 4. Improve Existing | 5. No System Change |
|---|---|---|---|---|---|
| **Modernization** | Highest | High | Moderate | Low | None |
| **System Change** | Highest (Enterprise Level) | High (Application Level) | Low (Localized Change) | Moderate (Application Level) | None (Process Change) |
| **Risk Severity** | Highest Risk | Moderate Risk | Low Risk | Moderate Risk | Risk of Falling Behind |
| **Funding Need** | Highest | High | Moderate | Moderate | Low |
| **Duration** | Long (Multiyear) | Moderate (1-2 years) | Moderate (2-6 months) | Moderate (4-10 months) | Not Applicable |
| **Benefits** | Highest | High | Moderate | Moderate | Little |
| **Popularity** | Low | High | Highest | Moderate | Low |
| **Success Rate** | Moderate | High | Highest | High | Not Applicable |

**Conclusion**

Each of the five types of transformation approaches has their own merits and applicability. Which approach works best for a client depends on the client's organization context and there is no single Silver Bullet for all situations. Clients can follow one approach for entire company or chose different approaches in different units (e.g. LOB, business functions). The best way to

determine the approach is to perform a due diligence phase to evaluate the engagement and come up with the recommendation & roadmap. Typical such assessment phase is 4-6 weeks long and involves all stakeholders associated with the modernization. It may be beneficial to leverage the experience / lessons learned from past programs done on similar functionality & technology.

**Reference**

1. Ulrich, William M, Legacy Systems: Transformation Strategies
2. Seacord, Robert C., Modernizing Legacy Systems: Software Technologies, Engineering Processes, and Business Practices (SEI Series in Software Engineering)
3. Mangalwadi,Vishal, The Legacy of William Carey: A Model for the Transformation of a Culture
4. http://doit.maryland.gov/SDLC/Documents/Legacy%20Transformation.pdf
5. http://adm.omg.org/legacy/ADM_whitepaper.pdf
6. http://searchcio.techtarget.com/definition/application-modernization
7. http://www.brianmadden.com/blogs/jackmadden/archive/2014/09/30/is-legacy-application-transformation-the-next-big-thing-in-our-space.aspx

**About the Author**

**Debiprasad Mukherjee** is a Business Consultant with Ericsson Global Services, with over 10 years of experience in IT solutions, analysis and consulting predominantly in Telecom & Healthcare industries. Debiprasad has extensive experience in business process management, automation, optimization & transformation programs. He has handled inception to implementation of 10+ different large scale process management & modernization programs. Debiprasad, an Electrical Engineering graduate & MBA, has published several articles/research papers in different leading international journals, magazines and forums across geographies.