POLARIS®

live your dream

The Importance of Adopting the Right Software to Reduce Technology Debt

Content

Introduction 1

The technology stack of a software application is decided during its inception phase. Numerous software technologies, each of various versions are available for the development of enterprise applications and software products. As the choice of technology is important in determining the time to market and Return on Investment (ROI) of the developed application, choosing its right version is equally important. Failure to choose the right technology or its version could lead to a debt which is a form of technical debt [1], called Technology Debt [2].

This paper uses Geoffrey A. Moore's Technology Adoption Life Cycle [3] to illustrate the roles that Technology Decision Makers (Architects or IT Specialists or Technical Leaders or stakeholders who are responsible for making technology adoption decisions) should play while adopting a technology in various phases of application development to reduce Technology Debt. It further suggests that as the application evolves, its technology stack should also evolve to reduce the Technology Debt and increase the lifespan and ROI of application.

2 Software Technology Innovations

Tools and technologies used to develop a software application evolve with time. The innovations in software technologies are classified as continuous and discontinuous innovations. Innovation of new technologies which change the way an application is developed is called discontinuous innovation. Introduction of AJAX, Spring, Struts or Object Relational Mapping (ORM) frameworks are examples of discontinuous innovation. Upgrades to a technology which do not change the way it is implemented are called continuous innovation. Release of Java SE 7 after Java SE 6 is an example of continuous innovation. New vendor implementation of a standard technology is also considered as continuous innovation. ObjectDB - an implementation of Java Persistence API (JPA), which is a technology standard is an example of continuous innovation.

Software technology providers release technology innovations as major or minor releases. A major release has significant difference in features than its preceding releases. A minor release includes fixes for defects found in its preceding releases. Every release has a version number associated with it.

Continuous innovations usually have backward compatibility with previous versions. Hence, implementing a continuous innovation in applications that are developed using previous versions rarely has a significant impact on the code, though there could be a considerable improvement in performance and resource utilization. Implementing a major release may require changes in the existing code as APIs in technology may get deprecated.

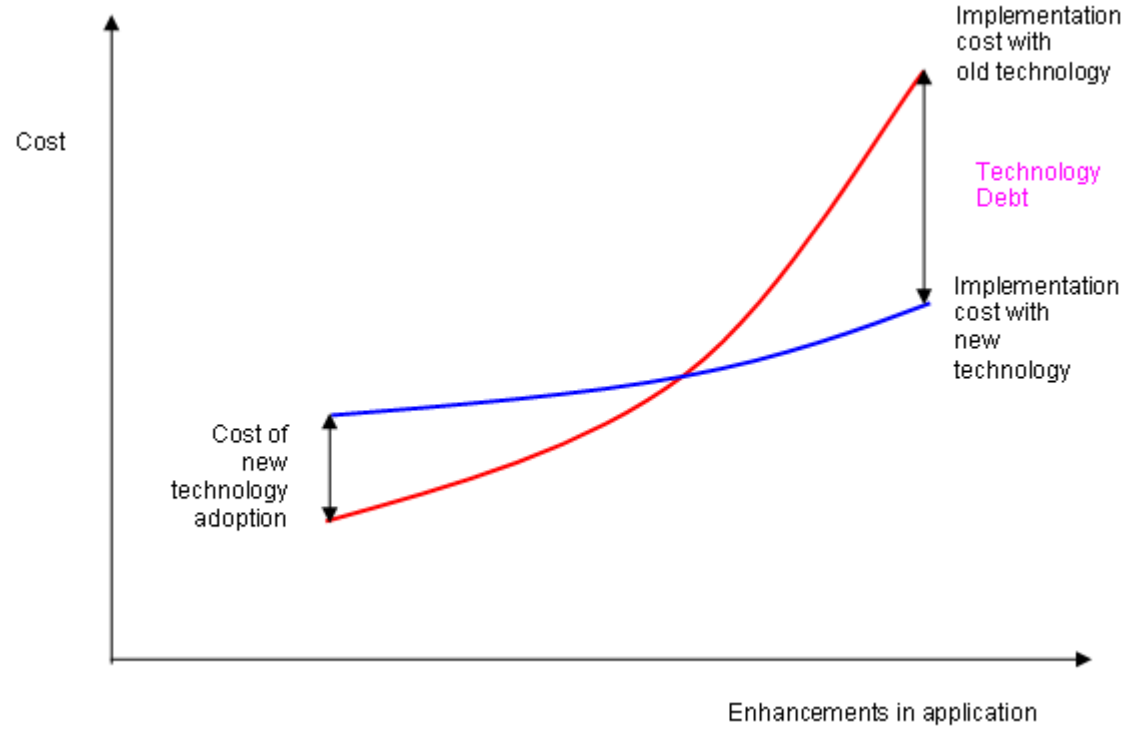
Continuous or discontinuous innovations in technology occur with the passage of time. Discontinuous innovations often provide cost benefit by reducing the development effort.

Technologies become obsolete with time. At times, a technology provider stops providing support for older versions. It could be a show stopper for ongoing development or it might pose a security threat. For example, using Java SE 6 or older versions for which the public support has ended could pose a security threat.

What is Technology Debt?

Ivan Schneider has introduced the term Technology Debt [2]. This paper elaborates it further.

As an application grows older, the technology adopted by it also grows old as succeeding continuous or discontinuous technology innovations are introduced in the market. As a technology gets stale, its business value diminishes as improved features get added in the succeeding innovations. The implementation cost of a functionality with old technology is always higher as compared to the implementation cost of the same functionality with succeeding technology innovations. This differential cost surpasses the cost of adopting the succeeding technology in the application as the enhancements in application with old technology increase. This creates a debt called Technology Debt and it increases as the enhancements with old technology increase. Figure 1 illustrates Technology Debt.



The Technology Debt can be calculated as follows:

Technology Debt = Implementation cost of enhancements done with old technology after new technology was adopted by mainstream market – (Adoption cost of new technology + Implementation cost of same enhancements with new technology)

An example of Technology Debt in software application is given below:

The cost of implementing a functionality in JDBC is greater than its cost of implementation in JPA as there is no need to write low level SQL queries in the code developed using the latter. If the application continues to use JDBC for enhancements, then some day the cost of enhancements done after JPA was adopted by mainstream market, will exceed the sum of cost of adopting JPA in the application at the time it was adopted by the mainstream market and the cost of subsequent enhancements in JPA. This difference in the cost is Technology Debt.

A Technology Debt is a risk and it cannot be completely eliminated. It can be reduced by adopting right technology innovations in applications.

4 How to reduce Technology Debt?

The Technology Debt can be reduced by adopting right software in applications. The Technology Decision Makers can take on adoption roles specified by Technology Adoption Life Cycle in various phases/types of application development.

5 Technology Adoption Life Cycle

In 1991, Geoffrey A. Moore had introduced Technology Adoption Life Cycle Model (Figure 2) in his book Crossing the Chasm. The model illustrates how a technology gets adopted in market with the passage of time. It divides the adopters of technology into the following categories: Enthusiasts, Visionaries, Pragmatists, Conservatives and Skeptics. Each category is important from the perspective of market dynamics.

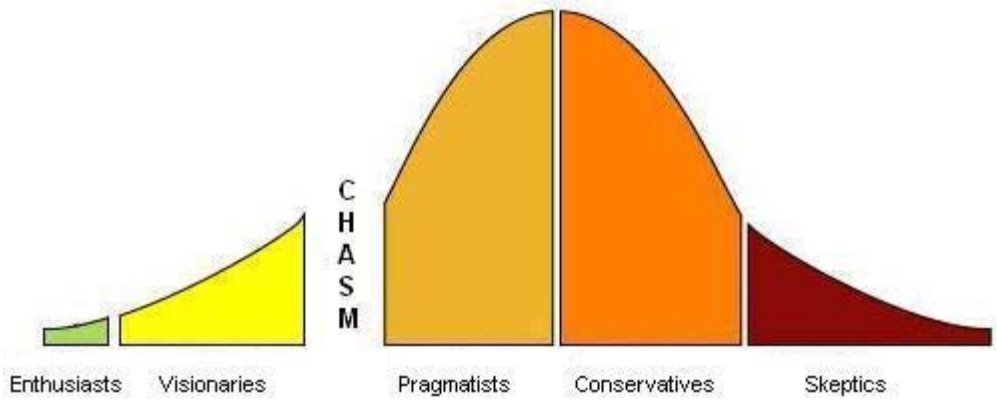


Figure2. Technology Adoption Life Cycle

When a new technology is introduced in market, the Enthusiasts are the ones to adopt it first. They are also called Innovators. This category of people, though not constituting more than 2.5% of the market, is constantly on the lookout of trying new technologies. They are ready to experiment with a new technology while it is still in nascent stage in market and thus take the risk of adopting a buggy one.

Visionaries, also called Early Adopters, foresee the potential of an emerging technology and are ready to take a leap by applying it to a strategic opportunity. This category constitutes of 13.5% of the market. They usually belong to executive class and are driven by a dream of a business goal.

Unlike Visionaries, Pragmatists or Early Majority, believe in making incremental progress instead of taking a quantum leap forward. They look for quality and adopt technologies which are fully developed. They need choice to evaluate the best technology and an alternative to fall back on when there is something wrong with it. Hence, they prefer to adopt standard technologies from established market players. This category is reasonably price sensitive and is willing to pay extra only if they are getting niche technologies. Pragmatists care for opinions about the technology from their peers. This category constitutes of 34% of the market.

Conservatives also called Late Majority are reluctant to try out a technology unless it is widely accepted in the market. They tend to invest in a technology towards the end of its life cycle. By the time they adopt a technology, it is mature and they get commodity level pricing. For every Pragmatist there is a Conservative. This category also constitutes of 34% of the market.

Skeptics or Laggards are the last ones to adopt a technology if at all they do. This category constitutes of 16% of the market.

In the model (Figure 2), there is a gap between each category. However, the gap between Visionaries and Pragmatists is considerably large than the gap between rest of the categories and it is called as Chasm.

This model is applicable to adoption of both continuous and discontinuous innovations in technologies.

Which technology gets adopted in which phase of its life cycle? It depends on the Technology Decision Makers – whether they are Visionaries, Pragmatists, Conservatives or Skeptics.

6 Software Technology Adoption in Applications

When a technology provider introduces a new innovation, it is generally showcased in events such as technology meetups, conferences, forums, social networking sites etc. It could also be implemented on small scale in internal projects by the technology provider or its partners. It keeps evolving on the feedback of Enthusiasts.

The Visionaries may pick up an innovation for implementation after initial positive response. These are the people who are well connected with leaders across industries and regularly attend conferences and seminars. They are active on technology groups on social networking sites and keep a close eye on events occurring in IT industry. Since Visionaries are usually outgoing people, their feedback is available publicly on blogs, interviews or social networking sites. The technology also evolves based on the feedback of Visionaries. Still, this is not a large scale adoption of technology. The technology has to overcome the chasm in order to get widely adopted in the software industry.

What happens if a technology never crosses the chasm? If a technology is not marketed properly or if it has any drawbacks or gaps or if a better technology becomes available in market, then it is possible that the technology is never able to cross chasm and thus never gets adopted by mainstream market. In such cases, it gets lock-in in projects if the Visionaries have already implemented it. For example, technologies such as EAD4J and JDO were never adopted by the mainstream market but few projects still use them. Visionaries have to bear this risk.

When does the technology migration happen for projects with lock-in technologies which do not cross the chasm? It takes time for any technology to get adopted by the mass market. There exists a time gap, when Visionaries decide to implement a technology and when it reaches the chasm. By the time it is realized that a technology would never cross chasm, it has been implemented already or is being implemented. Reverting to some other technology can have significant impact in terms of wastage of effort in learning the technology and redevelopment of deliverables. Usually, Visionaries decide to go ahead with the current implementation with a hope that the technology will be picked up by mainstream market sometime later. The Visionary may decide to switch to some other technology only when a gap is identified in the new technology during POC or application development. Thus, the technology migration rarely happens immediately for lock-in technologies which do not cross the chasm.

Though it makes sense to continue using such lock-in technologies which become obsolete with time, it becomes even harder to get developers skilled in them. Often the project managers have to consider learning curve for new developers while onboarding them into projects on such technologies.

Applications implemented in technologies which are never adopted by the mainstream market often become potential candidates for technology migration projects after some years.

What happens when a technology crosses the chasm and starts being adopted by software companies? The Visionaries who have adopted it are at advantage. They have already adopted a technology which other software companies are contemplating adoption. They are certainly at edge over others in competition in terms of reaping the benefits of technology, getting technology license in lower or discounted price and business advantage of being ahead of the herd. Thus, the Visionaries derive value not from the technology but from the strategic leap it provides.

Pragmatists are the people who are active on social networking sites and are connected with their peers but may not follow all events in IT industry very closely.

Conservatives plan to implement a technology only when they have read or heard enough about it. They do not follow social networking sites. They rely more on the word of mouth. These people usually stick with the technology they use and are against adopting new innovations unless it is absolute necessity.

Skeptics are reluctant for any technology change. They continue to use old technologies though there could be newer and better technologies. They may face difficulty in hiring resources to work on outdated technologies. They could also end up on spending up-front training costs to train new developers on technologies outdated in the market. They lose on the following advantages of adopting technologies which the other roles gain:

- Faster time to market by reduction in development effort
- Fixes for security threats
- New and/or enhanced features
- Less consumption of resources due to optimized features
- Availability of skilled manpower in the market (applicable only for Conservatives)

Whether a software technology is an Open Source Software or it needs to be purchased, is also an important factor in its adoption in applications.

An example of technology adoption behaviors is as follows:

JDK 8 is the new version of Java SE which was released in Mar 2014. Oracle had also provided its early access release. Enthusiasts were the first to try out new features of JDK 8 such as Lambda with the early release and provide their feedback on the web. Visionaries have started adopting it. Pragmatists will continue using JDK 7 until JDK 8 becomes stable and they start reading good feedback about it. Conservatives will continue using JDK 6. Skeptics will still use JDK 5 or JDK 1.4.

7 What role should Technology Decision Makers play in adopting a software technology to reduce Technology Debt?

The application development that takes place today in IT Products and IT Services industry can be for the following types of applications/phases:

- Proof of Concept (POC)
- New Application
- Application Maintenance
  - Enhancement
  - Ongoing Support

Proof of Concept (POC)

A Proof of Concept (POC) is an application that is developed to evaluate the feasibility of a new technology or a new feature of an already implemented technology or a new business requirement which has never been implemented before by a business unit and its gap analysis. A POC is often developed for new business engagements. It can be developed for internal stakeholders within an organization or external clients to demonstrate the working application. Since the purpose of POC is just to evaluate feasibility, time is a critical factor in its development. A POC is often developed using rapid application development and the practices followed in standard development life cycle such as code reviews may not happen.

Which technologies should be used in POC? It depends on the purpose of POC. If the POC is being developed for a new application or to evaluate the feasibility of a business requirement within an application, then the technology that enables rapid application development is preferred. If components of POC are to be used in actual application then technology which is the best fit for implementing it is preferred. Business units within organizations often prefer use of standard set of technologies in their projects. For example, some business units develop applications in Spring-Hibernate or some use .NET technologies in their projects. If the development of a business requirement does not demand use of a specific technology then it makes sense to use common technologies throughout the business unit since there is no skill gap and resources needed to develop POC can be easily made available.

A POC for implementing continuous innovation in an existing application is rarely developed unless a major code change is anticipated. For example, a POC can be developed for implementing Fork/Join Framework of Java SE 7 or analysis of migration from Hibernate 3.5 to Hibernate 4.0. For implementing discontinuous innovation in an existing application, the Technology Decision Maker within a business unit may develop a POC for evaluation and gap analysis.

Technology Decision Makers should play a role of a Visionary or Pragmatist for POC development.

New Application

A new application is developed for a new business goal or a business requirement that cannot be embedded within the existing application. It can also be built as a technology migration project for a legacy application that uses obsolete technologies. With time, the maintenance of legacy application becomes unmanageable due to its growing scope, lack of availability of skilled developers and technology limitations such as lack of supporting technologies or inability to implement a certain requirement. A new application may replace a legacy application entirely or partially.

A POC is often developed for a new application. The latter can be built from scratch or can reuse the components of POC.

For a new application, Technology Decision Makers should play the role of a Visionary or Pragmatist.

Application Maintenance

Once the initial version of an application or product is delivered, its maintenance phase starts. It consists of Enhancement and Ongoing support. In Enhancement, major or minor versions of an application are released. Ongoing support deals only with fixing bugs in a release. The fixes for defects resolved in ongoing support are delivered either as a patch or as a minor release.

Application Maintenance is a very important phase since majority of applications in IT Software industry today are in this phase. Also, this is the longest phase in the life cycle of an application.

Implementing a technology which is not yet adopted by mainstream market and thus being a Visionary could pose a risk of lock-in technology as mentioned above.

Enhancements could be major or minor. They can be done locally within a module or throughout an application.

Changing the existing technology stack is not advisable for minor enhancements. Technology Decision Makers should play a role of Skeptics for minor enhancements.

A major enhancement could be changing or implementing new features in one or more modules of an application or throughout the entire application. This change can be at local, non-local, application framework or deployment architecture level. Any change that is being implemented should add to the business value of application. An example of a major enhancement which may need change at the architecture level is to extend existing web application to support iPad and other mobile devices.

When a major enhancement needs new modules to be implemented which are loosely-coupled with rest of the application, Technology Decision Makers can play a role of Visionary or Pragmatist or Conservative and implement continuous or discontinuous technology innovation. This is a wide range since they have to evaluate multiple factors such as estimated future lifespan of the application, scope of enhancement, future enhancement expectancy, budget, costs, availability of developers, learning curve to implement a technology and deadline for the enhancement to go live before making a technology decision.

For all other major enhancements, Technology Decision Makers should play a role of Pragmatist or Conservative and implement continuous innovation.

Often, Visionary or Pragmatist or Conservative Technology Decision Makers recognize the need to adopt continuous or discontinuous technology innovations in the existing application to reap their benefits in future enhancements. They undertake the activity of technology adoption as a major enhancement in this phase.

Technology Decision Makers working on ongoing support should be Skeptics.

8 Summary

As the software technologies evolve and get adopted for application development by software companies, Technology Decision Makers (Architects and other stakeholders who are responsible for making technology adoption decisions), play an important role in their adoption. All adoption roles depicted in the Technology Adoption Life Cycle – Enthusiasts, Visionaries, Pragmatists, Conservatives and Skeptics - are important in the life cycle of an application as they are important for market dynamics. This paper explains how Technology Debt is incurred in Software Development and suggests the roles that Technology Decision Makers should play in various types/phases of application development to reduce it.

The following tables, Tables 1 and 2 summarize the roles that Technology Decision Makers should play while adopting a technology in these application types/phases and the innovation types that should be adopted by them.

Application Types/Phases	Enthusiast	Visionary	Pragmatist	Conservative	Skeptic
POC		✓	✓		
New application		✓	✓		
Minor enhancements					✓
Major enhancements as new modules		✓	✓	✓	
Major enhancements in existing modules			✓	✓	
Technology adoption as major enhancement in existing application		✓	✓	✓	
Ongoing support					✓

Table1. Roles to be played by Technology Decision Makers in various types / phases of application development

Application Types/Phases	Continuous	Discontinuous
POC	✓	✓
New application	✓	✓
Minor enhancements	✓	
Major enhancements as new modules	✓	✓
Major enhancements in existing modules	✓	
Technology adoption as major enhancement in existing application	✓	✓
Ongoing support	✓	

Table2. Innovation types that should be adopted in various types / phases of application development

9 References

1. Metaphor developed by Ward Cunningham - http://en.wikipedia.org/wiki/Technical_debt
2. Getting out of Technology Debt - http://www.enterpriseefficiency.com/author.asp?section_id=905&doc_id=231135
3. Crossing the Chasm – Geoffrey A. Moore

Polaris Financial Technology Limited

About the Author

Madhura Oak is a Project Manager at Polaris Financial Technology Ltd. She has over 12 years experience working on Java EE and XML technologies. Madhura has worked extensively on Project Management, Architecture, Design and Development of Enterprise Applications and Software Products for 6 years.

About Polaris Financial Technology Limited

Polaris Financial Technology Limited is a global leader in Financial Technology for Banking, Insurance and other Financial Services. With over 25 years of expertise in building a comprehensive portfolio of products, smart legacy modernization services and consulting, Polaris owns the largest set of Intellectual Property for a comprehensive product suite, Intellect® Global Universal Banking (GUB) M180. Intellect® is the world's first pure play Service Oriented Architecture (SOA) based application suite for Retail, Corporate, Investment banking and Insurance. Its acclaimed products, solutions and services enable unprecedented operational productivity for the global Financial Services Industry by Building, Maintaining, Expanding and Extending highly complex and Integrated Financial Technology Infrastructure.

The company has a global presence through its 40 relationship offices across 30 countries, 6 international development centers and 8 fully owned Business Solution centers. Polaris has a talent strength of over 11,500 solution architects, domain and technology experts. For more information, please visit http://www.polarisFT.com

www.polarisFT.com