



Compuware
UNIFACE®

CONTAINING AND MANAGING

TECHNICAL DEBT WITH UNIFACE



All day, every day, ministers, bankers and business leaders discuss the debt crisis, to the point where the objective of containing and managing debt seems a worldwide obsession. Here Ton Blankers, Product Manager, Compuware Uniface, reviews the implications of a special kind of debt – technical debt – and the options for containing and managing it.

EXECUTIVE SUMMARY

The idea of technical debt was first presented back in 1992 and refers to the need to go back and rectify the consequences of shortcuts, such as doing development in a 'quick and dirty' manner to satisfy business deadlines. Technical debt, like financial debt, not only impedes growth and adaptability, but also incurs financial costs and waste. Containing and managing technical debt is vital to business success and growth.

IT - particularly software - is being seen as a key enabler of business growth, adaptation and differentiation, thanks to developments such as the rise of social media; mobile device proliferation and sophistication; and on-demand, pay-as-you-go delivery models like Cloud computing. The rush to capitalize on new software and IT platforms can create aggressive time-to-market targets for software development teams.

Such time pressures can leave businesses with mounting technical debt as a result of corner-cutting, and the costs of putting matters right can wipe out any initial advantages gained from speed. Other causes of debt include inappropriate development platforms, including 'shadow IT' arrangements made behind the back of the IT function; productivity problems when developers struggle with new technologies; and poor documentation or version control, which hamper maintenance.

While not always a bad thing in itself – since sometimes speed really is more important than standards – debt must be contained as far as possible and, where it is unavoidable, must be managed effectively. Compuware Uniface provides an ideal platform for doing both.

It allows you to combine existing expertise with novel technology to avoid botches. It also helps you take systematic, rapid advantage of options like Cloud computing so that users are not tempted into shadow IT. Because Uniface models are self-documenting and self-explanatory, the development process is faster and maintenance is straightforward.

All these features let you avoid technical debt as far as possible. When, however, a debt must be incurred, Uniface helps you manage it effectively. For example, by using Uniface's multi-level APIs, Web Services support and extensive integration capabilities, you can easily reuse existing components in order to create next-generation, Service Oriented Architecture (SOA) based, enterprise applications. By leveraging features like these, companies can more effectively manage, and ultimately decrease, technical debt.

CHALLENGING BUSINESS DYNAMICS

In a tough global business market, austerity is a watchword for governments and many organizations during their buying cycles. If you are hemorrhaging money through unmanaged debt then not only are your staff unable to do their jobs, but you also have little room to maneuver for innovation and growth. You end up spending significant portions of your budget on basic maintenance rather than on delivering competitive edge.

This is not a new problem. More than 70% of all IT budget is spent on maintaining infrastructure and software. Over the past five years, the move to a virtualized, commodity hardware environment has helped reduce some of those costs, especially in the physical infrastructure layer. With Cloud computing, there is likely to be further cost reduction in this area. However, as costs are reduced, so are budgets. This means that other ways to reduce costs need to be found. One possibility is to reduce technical debt.



WHAT IS TECHNICAL DEBT?

Technical debt is a metaphor first developed by Ward Cunningham back in 1992 to help us all understand the consequences of doing things in a 'quick and dirty' way. When that happens, a debt arises in the sense that you will need to rectify matters at a later stage. Put simply:

Technical debt is what is created when you want to add some functionality to your application, or implement some capability in your infrastructure, and you choose to do so through a quick but dirty process that will make change harder further down the line or in the future. Avoiding technical debt would require a cleaner and more planned approach that would take longer to implement but would support future change and adaptation more easily.

So from a software development perspective, technical debt is created by things that you don't solve during your current development cycle, but rather push out to the next cycle, when you may or may not solve them. And, like financial debt, technical debt incurs interest in the form of additional work that must be applied to rectify or re-factor the initial quick and dirty fix in order to bring it into line with the rest of the designed architecture, achieve compliance or satisfy governance regulations.

If we were talking about a failure in financial controls leading to a loss of revenue, then most organizations would immediately look at what is required to tighten the way that they do business. But, because software is an intangible for many organizations, technical debt is often overlooked.

This is a mistake. IT is most organizations' biggest single budget line, and badly managed technical debt is no different to any other form of financial waste.

GOOD VERSUS BAD TECHNICAL DEBT

Is all technical debt bad? Perhaps surprisingly, the answer is no.

When we look to purchase goods, we make the choice to buy on credit or to buy with cash. We all know that the cash option means no extra interest payments or debt, and should be our preferred choice. The reality is that often we need something before we have the money so we use credit cards and carry a degree of debt. If we are disciplined, we pay off the credit card bill when it arrives, before accruing any interest. The same analogy can be applied to software delivery.

In the same way that credit cards and business loans often allow you to make large purchases that will pay for themselves, some technical debt can allow you to write software that will enable the business to react quickly to market conditions. In that situation, incurring technical debt is justified. It allows you to deliver software quickly to achieve competitive edge and market share, postponing the moment when you will bring that software into line with your standards and best practices.

However, in the same way that you would devise a business plan before buying on credit or taking out a loan, you need to have a process that allows for and pays back technical debt. The longer you defer repayment of the debt by delaying fixes, or the more you increase the debt by adding further workarounds, the harder it will be to pay it off. Worse still, like the loss of financial capacity, technical debt may eventually restrict your ability to respond to market changes and growth. Ultimately, technical debt can become a long-term drag on business agility.

If you want to make sure your technical debt is 'good' debt, the trick is to prevent those credit payments from getting out of control, which means repaying the debt regularly as part of your development process. It is important to have metrics and tracking mechanisms to identify technical debt, and processes for managing and reducing it. Let's take a closer look at how technical debt happens, before discussing ways to get it under control.

THE ROAD TO TECHNICAL DEBT

With all the time in the world and no commercial pressures, we can all produce gold-standard software. Better still, we can take the time to learn about new technologies and acquire the skills to implement them and integrate them with existing technologies and systems. Unfortunately, this is a luxury companies do not usually have, and when the reality falls short, technical debt can occur. Below we discuss some of the reasons.

TIME TO MARKET

The need to reduce time to market (TTM) is often an excuse for quick and dirty software development. The business believes that it needs the software fast, even if it means accepting some weaknesses, and developers know that they are under pressure to deliver. All this is more likely than ever to happen in today's volatile market, when organizations expect their IT functions to help them to respond quickly to changing conditions and goals.

When we produce software to meet TTM requirements, inevitably we build in problems. The resultant bugs are accepted as a cost of doing business. This is the software equivalent of buying on credit. As well as contributing to other quality defects, TTM pressure is one of the main culprits behind technical debt. But it is not the only one.

INAPPROPRIATE DEVELOPMENT PLATFORMS AND PROCESSES

The choice of development platform and working practices can also cause technical debt to accrue. Problems can include a sub-optimal choice of process methodology, lack of defined quality processes, poor requirements gathering, project scope creep, ineffective project management or leadership, poor training, and shadow IT.

Shadow IT deserves particular consideration. It occurs when multiple groups try to run their own IT solutions without coordinated management. Typically this happens when business units, frustrated by what they perceive as slow delivery from in-house IT teams, look outside the company for alternative solutions. For example, they may choose Software as a Service solutions for email or office applications. The challenge comes when IT must manage and maintain solutions that were commissioned without their initial involvement.

PRODUCTIVITY LOSS

Another significant factor is the absence of an application platform and framework that allows the team to take easy advantage of new technology, update to the latest standards and integrate with a host of application technologies already in play. Understandably, software teams want to adopt emerging and maturing software trends, and indeed they need to do so if they are to satisfy the business's demand for increased agility in the longer term.

However, there is a productivity loss whenever teams have to spend time and resources gaining knowledge to implement new software technology and application models. That is particularly true if it means they are not leveraging investments in their existing development skills and platforms. This productivity loss creates a debt that can be hard to claw back.

MISMATCHED INFORMATION AND POOR BACKWARDS COMPATIBILITY

Software development is a bit like building a house. The people who initially build the house or application are often different from the ones who will maintain it, and it is the maintenance team who will have to cope if, for example, the builders left inadequate documentation about the exact make and model of the water pipes.

Like new houses, applications are delivered with technical descriptions, which are supposed to include not only the original design but also any changes made during development. However, the reality can fall short. With applications built using UML modeling tools, changes are often not reflected in the documentation of the model – and that creates a technical debt. There can be similar problems with Java or .NET applications, where new framework versions may not be compatible with older ones.

UNIFACE CAN HELP YOU MANAGE TECHNICAL DEBT

Having seen how technical debts arise and can spiral out of control, let's consider some simple ways to ensure that your technical debt is all 'good' debt, and that it is well managed. Doing so does not have to impose an additional financial burden on already overstretched IT budgets. It is possible to reduce and manage technical debt by applying the right tools and processes, plus a bit of common sense.

It is important to accept that incurring a debt is sometimes the right choice. The trick is to strike the right balance between speed and innovation on the one hand, and efficiency, quality, stability and consistency on the other. If you slow down development to ensure that there are no bugs and that the software is bombproof, then delivery will come too late for the business. Go too fast and you will waste huge amounts of time, effort and money. Of equal concern is the loss of internal customer confidence in the software.

Technical debt cannot be tackled through deployment of point solutions. What is required is a comprehensive development environment in which technical debt can be assessed, managed and contained. This environment should allow your software development teams to adopt the latest delivery models and technologies in order to provide the business with the agility it requires. But at the same time it should allow them to make use of their existing skills: doing so will help to minimize the technical debt that can arise from trying out the new.

Compuware's Uniface gives you all that and more, as this section explains.

DEBT CONTAINMENT THROUGH A MANAGED PLATFORM

The challenge of getting the development environment right should not be underestimated. Part of the challenge is to support integration of new applications and technologies with those already in place. Without a strong integration strategy and framework, along with the means to maximize productivity benefits of development skills in which you have already invested, any gains from implementing new software advances will be limited.

Uniface's integration framework and model-driven development platform promote rapid development of high-end, complex, enterprise applications that interface with multiple technologies and application architectures (mobile, mainframe, SOA, RIA, Windows, JEE and .NET, Web2.0 and Cloud). The model-driven development approach provides a managed development framework that hides much of the complex plumbing required to support new technologies and delivery models like Cloud computing and SaaS solutions.

Uniface offers an elegant solution to the problem of combining existing skills and knowhow with the newest technologies and architectures. By using it, you can achieve high developer productivity to satisfy TTM requirements, while containing and minimizing the technical debt that can arise from quick and dirty technology integration. Its robust integration framework and proven roadmap for supporting new technology trends mean that you can innovate with confidence. With the recent addition of WSDL, XML Schema and complex data types, Uniface developers can use the latest Web Services implementations without an extensive learning curve.

DEBT CONTAINMENT THROUGH PRAGMATIC TECHNOLOGY INTEGRATION AND SUPPORT

We know that customers are just as concerned with the question of when to begin incorporating the latest software advances as they are with the question of how to do it. They need frameworks and development platforms that allow them to feel confident in taking advantage of optimized implementations that incorporate best practices from market adoption and maturity. Therefore, it is not only important that development platforms keep abreast of important software advances and application models, but that they do so through a pragmatic and realistic integration and support roadmap.

The Uniface platform has a strong product architecture and roadmap for supporting new application technology advances and delivery models. The current release offers increased support for legacy modernization, particularly for Unicode sorting, encryption and decryption, workflow integration and deployment to the Windows 7 operating system. There is also support for the development of richer and more engaging web applications through the new Rich Internet Application (RIA) technology features. Included are templates that enable users to implement RIA-based technologies such as AJAX (Asynchronous JavaScript and XML) productively, and create RIA-based applications through Uniface's programming model.

The improved SOA and Web Services support that Uniface delivers, along with services and utilities for automated performance analysis, test-driven development and continuous integration, will allow customers to take a modular approach to getting ready for Cloud computing. The RIA and Cloud computing facilities leverage Compuware Uniface's strength in developing applications for distributed computing and the company's investment in SOA and Web Services infrastructure.

Because you can take advantage of new technologies rapidly with Uniface, you may be able to reduce the temptation to short-circuit the process by, resorting to shadow IT arrangements, with the technical debt that they entail.

DEBT CONTAINMENT THROUGH INTEGRATED QUALITY MANAGEMENT

It is important to make sure that developers follow the right naming conventions, and that appropriate services are used consistently throughout all phases of software development. Mistakes in these areas make application maintenance more expensive and more sensitive to programming errors. Constant monitoring of software implementations against the appropriate standards and guidelines is therefore a must if technical debt is to be minimized. To achieve it, you need quality management facilities that are integrated with all phases of the software development lifecycle: Uniface gives you that integration.

DEBT CONTAINMENT THROUGH EFFECTIVE MAINTENANCE

Unlike UML models which often get out of synch with the application, Uniface's graphical models of applications are automatically kept up to date, however rapidly your development takes place. The models also allow maintenance teams to gain an immediate understanding of how an unfamiliar application works. Finally, Uniface applications are written using far fewer lines of code than traditional techniques create, which reduces the maintenance workload.

In contrast with Java or .NET, Uniface avoids the use of external frameworks, different versions of which could become incompatible. Uniface does this by implementing all new functionality through Application Model and Uniface Proc Language extensions. In this way, Uniface, as it always has done, guarantees upwards compatibility, so that developers don't have to worry about technology version differences.

DEBT ASSESSMENT AND ANALYTICS

Strong intelligence and analytics will help you assess the likelihood of accruing technical debt, and your capacity to contain and minimize it. A good starting point is to look at the information you already have, and at what you can do to get more information out of your existing systems.

Effective collaboration between the different stakeholder teams within the IT organization, and with the business sponsors, is also vital. Any managed development environment should dovetail with production and operational management systems to ensure continuity of debt management and containment.

Compuware Uniface is highly integrated with several other Compuware products (e.g. Compuware Gomez). This integration means that we can provide comprehensive management capability and performance monitoring facilities, allowing customers to address bandwidth and deployment issues and assess the quality and efficiency of their application solutions.



DEBT MANAGEMENT: PRACTICAL GUIDANCE

For those who would like to apply the debt management concept to their business, we recommend a few simple steps.

Review your whole approach to development from a debt point of view. Accept that some technical debt is an inevitable part of development, and build in processes to deal with it. Do not simply consign responsibility for debt to the software maintenance team: this response suggests avoidance rather than resolution.

DEBT MANAGEMENT: PRACTICAL GUIDANCE (continued)

The choice of methodology is an important step in managing technical debt. Waterfall and Agile approaches each offer advantages for debt management. Designating a development cycle as a debt management phase offers the opportunity to contain debt. By auditing the application right after the coding phase, software mistakes can be resolved earlier, which is much more cost-effective than fixing them, for example, in the final system test phase. However, for long-term methodologies like Waterfall, the development cycle can take many months to complete, causing the debt to act as a drag on all subsequent development. The relatively short iterations of Agile development, on the other hand, mean that technical debt can mount up rapidly, but by the same token there are more opportunities to resolve and contain it quickly.

There is no point trying to manage your finances if another family member is a shopaholic; therefore, we recommend two steps that will help everyone in the company to understand the risks of technical debt and help to manage them. First, put rules in place to minimize the risks of shadow IT arising from 'unofficial' use of Cloud services. Second, ensure that the business units are actively engaged, together with the IT function, in the development of key software components. The first step will save money across the board, as well as reducing the company's risks (such as that of data loss), while the second step will commit the business units to the whole process.

Consider delivery models that allow you to adopt and manage technology advances quickly. One of the benefits of Cloud computing is the ability to take advantage of new technology updates and application models and gain access to skills and computing resources on demand at an affordable price. Properly used, Cloud computing can help customers to contain their technical debt charge, prototyping new technologies without heavy upfront capital expenditure and resource acquisition.

Let Compuware Uniface also help you manage and decrease technical debt by taking automatic advantage of technology advances such as Cloud deployment. The Uniface Proc Language works at a high level of abstraction, and new technologies are added to the language through straightforward and easy-to-learn commands. In addition, the Uniface Run Time system takes care of time-consuming technical 'plumbing'.

Developers can avail themselves of many technical benefits without any additional programming. For example, Race conditions are automatically taken care of in Web applications, as are decreases in server load, because Uniface validates data on the client instead of the server. And the central Application Model greatly reduces code redundancy, minimizing coding errors and ensuring cost-effective maintenance.

Compuware Uniface will help customers achieve staged delivery of Cloud computing focused services, ahead of the launch of Uniface 10.0. For further insight into Cloud computing and Compuware Uniface's value proposition, download our [Cloud computing executive report](#).

CONCLUSION

Getting to grips with technical debt is a business imperative that is as important as controlling departmental spending. And just like departmental waste, it can be difficult to identify technical debt until you start looking for it.

Customers can trust Compuware Uniface to continue helping them to take on new technologies easily and securely without the botches that would cause technical debt to spiral. The modeling abstraction layer and managed development environment provides long-term maintainability and allows customers to take advantage of technology advances within existing application investments. We build the new functionality you need into the Uniface Proc Language, so your developers don't have to learn or use multiple frameworks from different vendors and sources.

The Uniface platform also offers a viable value proposition for organizations with legacy environments in need of modernization and optimization. Your skilled development teams can take advantage of new business and delivery models like Cloud computing, and new application models like RIA, without unduly compromising productivity and hence adding to the technical debt.

The Uniface Professional Services organization can provide insight into existing systems by checking application consistency and component dependencies. The reports they'll give you, combined with Uniface's Web Services functionality, will enable your applications to be effectively modernized so that you can get the full benefit of SOA.

But just as importantly, the Uniface development platform provides a framework to manage and contain technical debt for application development and technology integration, at both strategic and tactical levels.

Enterprise application development for the cloud... by design

WORLDWIDE



Compuware Corporation World Headquarters
One Campus Martius, Detroit, MI 48226
Phone: 313.227.7300 | 800.521.9353 | Fax: 313.227.7555
compuware.com

All Compuware products and services listed within are trademarks or registered trademarks of Compuware Corporation.
All other company or product names are trademarks of their respective owners, © 2011 Compuware Corporation

