Bloor

Spotlight

Spotlight Paper by Bloor
Author **David Norfolk**
Publish date **November 2021**

Refactoring monolithic systems an emerging business issue

A modern microservices architecture, on hybrid cloud, is the best basis for continuing evolution. The question is, how do you get there, without impacting your business effectiveness on the way?

Executive summary

An issue increasingly facing companies is the need to move from older, monolithic, but still business-critical and effective systems, to modern microservices-based architectures, supporting mutable businesses in a constant state of evolution. This might not be necessary if these systems never changed (you'd just provide an API) but mutable businesses need to be agile – and a modern microservices architecture, on hybrid cloud, is the best basis for continuing evolution. The question is, how do you get there, without impacting your business effectiveness on the way? Such older systems are often the hardest to modernise, partly because they are often critical to the business (and so, change is high risk); and partly because they have probably been *"optimised"* and heavily maintained, which can make them harder to understand.

IBM describes a modernisation process for Websphere applications, together with supporting tools, some of which use the latest Augmented Intelligence technology. As well as modernisation, this process helps organisations to decide when modernisation or (less often) application retention or replacement is appropriate. The usual goal is to support mutable business evolution, using hybrid Cloud technology, with everything running on the most appropriate platform for the business – but with as little waste of existing investments as possible. Although this Spotlight is couched in terms of IBM WebSphere and WebSphere Liberty, much of it is of general application.

This article is intended for the owners of large monolithic technology platforms (mainly, but not exclusively, IBM WebSphere) that are candidates for a low risk modernisation approach, and who want to be able to analyse a whole estate and assess and manage the impact of modernisation on the whole business both before and during their modernisation journey. It is also for developers who may be involved in the process, and for business managers who want to stay up-to-date on possible evolution strategies for the existing platforms on which they work.

It will also interest GSIs (Global System Integrators), interested in (primarily Java) modernisation generally, not necessarily targeting Liberty (although IBM believes that Liberty is the best application runtime for modern containerised workloads). IBM generally doesn't charge for its modernisation tools separately, they are part of WebSphere, but GSIs could potentially *"white label"* them for wider applications.

"An issue increasingly facing companies is the need to move from older, monolithic, but still business-critical and effective systems, to modern microservices-based architectures, supporting mutable businesses in a constant state of evolution."

The modernisation issues facing today's businesses

> **Many older applications are *"monolithic"* – that is, they are a single-tiered application, combining user interface, business logic and data access into a single application running on a specific platform.**

Most organisations today are considering whether they can take advantage of modern, modular, cloud environments, whether public or private. Often, organisations think primarily of cost savings on the Cloud, but as their Cloud vision matures, they realise that increased Mutability – the ability to evolve rapidly in response to a changing business environment – is a more significant benefit for the longer term. Some modern, modular, applications are easy to migrate to Cloud but many older, still business-critical, applications, aren't, and will require some degree of modernisation. Many older applications are *"monolithic"* – that is, they are a single-tiered application, combining user interface, business logic and data access into a single application running on a specific platform, usually in-house. Such applications don't translate well into modern cloud-services platforms. They perform a complete business task, from beginning to end, supplying every service needed for that task and often reside in a business silo instead of supporting the business as part of a holistic automation platform (although this is actually an extreme, such applications may make some use of services and support some links to other systems, but the stand-alone system represents the general picture). They can work well, until you try to change them, and they do some things (like synchronous transaction processing, where the data is kept consistent in real-time, regardless of transaction failure) more effectively than the Cloud finds easy (data in many Cloud applications may be inconsistent at any point in time, you may base processing on something that turns out to be incorrect, and make changes that no longer make sense, although the data eventually becomes consistent over time).

Faced with these issues, organisations have 3 choices, each with different cost, value and risk profiles. The choice has to be made based on a good understanding of the characteristics and usage of the candidate application:

- It can stay on the original platform, which may sometimes be the best choice (if it has a very limited life in front of it, perhaps), but which brings increasing risk as the platform falls further behind the IT environment generally. In any case, you should upgrade to supported levels of technology if you haven't already (this itself brings some risk, as behaviours may change after a simple recompile), for example, if source code is missing or out of date or the operating system services have changed).
- You can build a new, usually Cloud-native (designed 100% for Cloud) replacement application from scratch, which is likely to deliver the expected Cloud benefits (but don't overlook the possible cost and resource issues with validating the behaviour of the new application).
- You can modernise and usually refactor, the existing application, as described in this paper, exploiting your existing investment to produce a modern, modular, Hybrid Cloud application, which you can deliver incrementally (thus delivering value faster, and at lower risk).

Moving a monolithic application to an asynchronous platform, of many discrete interacting parts, isn't trivial, but in return, organisations hope to achieve more innovation, with faster time-to-market, better customer experience, better use of IT resources and better protection of sensitive data.

This sort of modernisation is a journey, a state of evolution in response to changing business. It is not *"rip and replace"*. Ripping out one platform that you rely on and replacing it with a new platform that you haven't relied on before and must exhibit exactly the same behaviours (perhaps with some well-documented changes) is seldom as cheap as expected and often also turns out to be far riskier than anticipated. The initial stages of modernisation may look like a discrete *"digital transformation project"*

but, in fact, it will be a program of related projects, going forwards in an Agile fashion – digital transformations, plural – and must be treated as such. Also, especially in large organisations, different departments or geographical locations may be on different stages of their modernisation journey at the same time. Large organisations, or parts of them, cannot change in zero time. You may be very clear as to why you don't want to be where you are, and very clear on the benefits of where you want to be at the end of your journey, but it is important that you don't go out of business on the way. Evolution proceeds incrementally, and each increment must be viable, or the evolutionary journey stops dead (and if it is working, it goes on practically forever).

Why should you care?

You should care about modernisation, because not doing so brings increasingly hard-to-address risk. Assuming that your un-modernised platform does its job well now:

- Vendor support for your platform may disappear or get more expensive.
- Hiring or training platform expertise may get harder or more expensive.
- Your unmodernised platform becomes a silo surrounded by modern technology – and silos are bad for morale and communications.
- Improvements to customer user experience may become harder to provide.
- *"Satisficing"* (taken from Nobel prizewinner H. A. Simon [***https://en.wikipedia.org/wiki/Satisficing***]) IT resources will get harder, because integration with older technologies will become increasingly difficult.
- It will get harder, with silos, to maintain a holistic approach to security.
- Eventually, your platform will stop doing its job well and modernisation will then probably make sense anyway.

Nevertheless, although modernisation matters, you should also care about maintaining choice and care about being able to modernise at your own pace. You also need to be aware of the possibility of evolving into lock-in with a particular cloud vendor. This is a big topic, which won't be covered here, except to say that Bloor thinks that you should target Hybrid Cloud, although there is more to avoiding Cloud lock-in than just that.

What are the benefits of modernisation?

The benefits of modernisation, as opposed to *"rip and replace"*, largely come from not wasting previous investment, as far as it still makes business sense. In more detail, you can *"mine"* existing applications for useful coded logic, which can be reused in modernised systems. Then, if older systems modernise through evolution you can:

- minimise disruption to the business;
- avoid reinventing the wheel;
- reduce (not eliminate) the need for regression testing to prove that behaviours haven't changed in unwanted ways at each stage of modernisation;
- reduce retraining requirements; and,
- preserve the freedom (within reason) to modernise at your own pace; and,
- reduce risk by delivering in *"minimal viable increments"*, which reduce the scope of impact of errors and enable the identification of possible mismatches between the automation and the business before they get too large.

Once systems have been modernised effectively, you should expect to significantly decrease the time to deliver new and innovative services; and to redefine the quality of customer interactions with real-time interactive services. You should also expect to optimise or satisfice your use of IT resources and to manage (reduce) cost and complexity, thus gaining competitive advantage. Modernisation should also allow you to improve security and the protection of sensitive data, by eliminating silos and improving communication. Security is a holistic thing (it is only as strong as its weakest link) and islands of unmodernised technology risk being left out of the security environment when it is enhanced in the face of emerging threats.

The benefits of modernisation ,as opposed to *"rip and replace"*, largely come from not wasting previous investment, as far as it still makes business sense.

> **Modernisation should not be a purely manual process these days. Human input will be needed to resolve issues and decide on priorities but emerging AI (Augmented Intelligence) tools are making modernisation both less expensive and more effective.**

Modernisation with assistive refactoring technology

Modernisation should not be a purely manual process these days. Human input will be needed to resolve issues and decide on priorities but emerging AI (Augmented Intelligence) tools are making modernisation less expensive, more effective and less error-prone. We will concentrate on one IBM product, Mono2Micro [*https://www.ibm.com/uk-en/cloud/mono2micro*], in detail, as an example; we'll also mention other IBM and Open Source tools below. Mono2Micro is the result of an extensive beta program [*https://www.ibm.com/cloud/blog/announcements/ibm-mono2micro*]), and uses AI to help you automate the refactoring of existing monolithic code (without changing its external behaviour and semantics) into effective and more maintainable microservices. A microservices architecture structures applications as a collection of services that are highly maintainable and testable (because they are small and mono-functional – cohesive – at some level); loosely coupled; independently deployable; business-focussed; and, owned by a small team. A microservices architecture makes rapid, frequent and reliable delivery of large, complex applications easier for developers – possibly at some cost in integration and communication complexity. It also makes evolution of the technology stack easier. microservices are also often (but not necessarily) associated with containers – think separate units of code – orchestrated or managed with Kubernetes [*https://kubernetes.io/docs/concepts/overview/what-is-kubernetes/*].

Incremental modernisation, in stages (with each delivering business value), is less disruptive and less risky:

1. Move to a modern, cloud- and container-optimized runtime such as Liberty
2. Move to a Kubernetes-based platform (or equivalent)
3. Modernise your architecture by refactoring your applications into individually deployable and scalable microservices)

An organisation might want to reverse the order of the first two modernisations, if it has already implemented a containerised platform with something like OpenShift. IBM Cloud Transformation Advisor (see below, in *"Other issues and supporting products"*) can help with the first stage by helping you to understand and assess existing applications and deployments, to identify areas requiring basic remediation or more intrusive refactoring, to estimate the effort that will be required; and, it will generate configuration information to help with the migration.

IBM Cloud Transformation Advisor also helps with **Stage 2**, the move to a Kubernetes platform. You are strongly recommended to keep your options open by using a multi-Cloud-managed Kubernetes solution such as OpenShift. There is another tool to help with this stage, the Open Liberty Operator (see below, in *"Other issues and supporting products"*), which helps with Liberty deployment and management in Kubernetes environments. This operator works with both WebSphere Liberty and Open Liberty in Kubernetes-based platforms that support the Operator Framework, which includes OpenShift.

Don't consider just containerising your traditional application servers, without moving to something like Liberty, as they were never designed to optimally support containerisation, modern DevOps practice or 12 factor [*https://12factor.net/*] cloud applications.

Stage 3, modernising architecture to deliver optimal scaling and cloud-native agility, is where the main subject of this Spotlight comes in. We will talk about the IBM product, Mono2Micro, but we'd anticipate that any application modernisation for Cloud, even without WebSphere, would need a similar approach.

Mono2Micro

The goal of Stage 3 is to enable Agile delivery of individual service updates through continuous integration, continuous delivery pipelines and DevOps – so you won't have to deliver (and test) a whole monolithic application. This will

also allow you to scale individual services in response to changing workloads, so as to optimise cloud usage and cost.

You will need to make code changes – or *"refactor"*. For instance, as we have already mentioned, if you are used to the rock solid transaction processing associated with older *"systems of record"*, decoupling data from business logic has implications – absolute consistency is hard to maintain in real time and many asynchronous, loosely-coupled Cloud systems rely on *"eventual consistency"*. A whole transaction is not locked until it all completes or fails (if it fails it all backs out and data consistency is maintained), as was usual in monolithic systems. Instead, microservices are allowed to update data individually, so one may act on information which a parallel micro-service may decide is invalid and removes – by which time another micro service has already processed something based on the changes made by the first micro-service. Back-out, as understood by transaction processing, is impossible, although eventually the data becomes consistent, if you code things correctly, taking advantage of well-known code patterns where appropriate.

Stage 3 is facilitated by the new AI-powered refactoring tool called IBM Mono2Micro [***https://www.ibm.com/uk-en/cloud/mono2micro***], which can analyze an existing monolithic application and suggest ways in which it can be cut up into microservices. You can do this incrementally, but don't forget to break up the data as well as the business logic – your new microservices are still coupled if they all reference a single data store.

Fundamentally, modernisation (which, in large part, involves breaking up a monolithic application into microservices) depends on discovering common code, which can be reused in a micro-service; and on discovering and dealing with dependencies (and removing dependencies that are dysfunctional). Mono2Micro is an intelligent tool which does more than just recognise patterns it has been told about. It can facilitate:

- Runtime tracing, to elucidate the dynamic behaviour of a monolithic application;
- Relating areas of code to business use cases;
- Establishing call/invocation volumes;
- Understanding classes and the business objects they represent;
- Understanding the clustering of calls/invocations;
- Understanding of code structure and application data dependencies (such as containment), minimising the need for code rewrite;
- Making recommendations, for manual implementation (the human developer, thankfully, remains in control);
- Reasoning about potential utility services based on code understanding;
- Identifying potential "dead code", which is a big issue, since it is effectively untested and, in future, a changed data input or code error could activate it, with dysfunctional results. It also increases code complexity;
- Code generation – not particularly for production use, but to deliver a working scaffolding for experimental refactoring; it also assists with the reuse of test automation and the refactoring of tests designed for the monolithic application.

When modernising applications, realistic and achievable estimates matter – partly because this helps with managing expectations – for a well-managed incremental journey to a more modern, Cloud-native runtime (that is, Liberty). Mono2Micro helps with this, working outside-in, focusing on the business cases and figuring out how to modernise around them, whereas Transformation Advisor works inside-out, starting with the existing technology artefacts and considering what needs to be adapted.

IBM Mono2Micro is delivered as part of the IBM WebSphere (IBM WebSphere Hybrid Edition) lead product, for no extra cost, but it could also be *"white labelled"* (and developed further) for/by IBM service providers and partners.

When modernising applications, realistic and achievable estimates matter – partly because this helps with managing expectations.

> **...in the old days, you paid for a product even if no one was using it (*"shelfware"*); these days if you aren't using something you don't expect to have to pay for it (or, at least, you can cancel the subscription).**

Other issues and supporting products

Licensing

Licensing is something to think about when modernising application suites. Older, monolithic, application platforms often have associated enterprise licences or site licenses; modern technology is more likely to have subscription-based licensing or even *"pay per use"* licenses. The modern licenses are better because they tend to share risk between customer and vendor more equably – in the old days, you paid for a product even if no one was using it (*"shelfware"*); these days if you aren't using something you don't expect to have to pay for it (or, at least, you can cancel the subscription). Licensing can be complex and this is not the place to go into it in more detail. Just make sure, when modernising, that you have the most flexible licenses available for the applications being modernised (unless there is very good reason not to); and that you get the best (most flexible, least restrictive, most cost effective) licenses possible for any new software. Also, don't forget that Open Source software, which may well be a part of a modernised platform, isn't licence-free software – make sure you read any Open Source licenses and think about what they commit you too (get expert advice if you are not used to Open Source). You might also want to check that you have licenced support, if you are using Open Source for business critical applications.

Useful products for IBM Websphere modernisations

- **IBM Websphere Automation** [*https://www.ibm.com/uk-en/cloud/websphere-automation*] allows teams to automate operations, so as to quickly unlock value with increased security, resiliency and performance. It can can help reduce costs and increase ROI. Optimising operations, so you can respond to incidents efficiently, and promote stronger security of the IT estate is an important aspect of modernisation.
- **IBM WebSphere Liberty** is a Java EE, Jakarta EE and MicroProfile low-overhead Java runtime, designed for cloud-native applications and microservices. WebSphere Liberty was created to be highly composable, to start fast, use less memory, and scale easily. A Total Economic Impact study from Forrester Research suggests that Liberty can increase developer productivity, simplify administration and maximize infrastructure utilization. It shares the same code base as the Open Source Open Liberty [*https://openliberty.io/*] runtime, which provides additional benefits such as low-cost experimentation, customisation and seamless migration from open source to production. The Open Liberty Operator [*https://openliberty.io/docs/21.0.0.9/open-liberty-operator.html*] helps with Liberty deployment and management in Kubernetes environments. This operator works with both WebSphere Liberty and Open Liberty in Kubernetes-based (containerised) platforms (including OpenShift) supporting the Operator Framework.
- **IBM Transformation Advisor** [*http://ibm.biz/cloudta*] is used to analyse on-premises workloads in preparation for modernisation. It assesses the complexity of the applications to be modernised, estimates the development cost of moving to the cloud, recommends the most appropriate target environments and identifies the specific code changes necessary. It helps with Identifying common code, which can be reused; the efficiencies associated with this can provide an immediate benefit to help pay for the modernisation process; and such code can also provide a good basis for future refactoring.

A typical modernisation use case

IBM reports that modernising its runtime with Liberty enabled a leading US healthcare provider to optimise its resource usage by 75% and reduced infrastructure footprint by 50%. This is the sort of thing that provides the incentive for modernisation.

Looking at a different Use Case in more detail:

Modernisation of transportation client's application estate

Risk Factors

1. Competition from organisations that have already modernized and automated for greater efficiency and agility.
2. Large, legacy, business-critical applications that don't easily accommodate automation.

Actors

IBM partner, modernising a public transit provider in Europe.

Priority

Essential to the partner's success

"As soon as we start delivering some first candidates of the applications to be modernized, they notice the advantage. They see faster deployment times, less impact. No long maintenance windows to deploy a new application" – Partner's Co-founder.

Status

In production.

Challenge

This business partner has many clients looking to automate their IT. This recent case of a public transit provider in Europe is a good example of this heavy demand from large organizations looking to modernize and automate for greater efficiency and agility, which are being held back by their large, legacy, business-critical applications, which are hard to modernise manually.

Consequences

Using IBM Transformation Advisor to analyze 65 legacy applications and understand the containerization effort, this partner used the Liberty operator to automate application deployment to OpenShift, thus reducing full deployment cycle time for its client from up to a month to about an hour. After modernizing the first application, the rest took relatively little work.

Key metrics:

- Deployment times reduced by over 99%.
- Developers get more done in less time.
- Ops teams gain greater support in ensuring stability.
- Business leaders are happy with how fast they can deliver new applications and features.

Solution Components:

- IBM WebSphere Hybrid Edition.
- IBM Cloud Transformation Advisor.
- IBM WebSphere Liberty.

Bottom line

If you want to go into this issue in more detail, in an IBM Websphere context, there is an IBM article [*https://developer.ibm.com/articles/modernize-your-valuable-java-applications/*] that is well worth reading A lot of what it says is entirely general:

"The approach you take to modernization can vary for each application and should be based on a number of factors:

- *The life-expectancy of the application*
- *Business needs of the application (for example, innovation, agility)*
- *Cost, whether the aim is to reduce cost or to invest for growth*
- *Technology skills and code reuse*

"While there is a lot of valid buzz around using a microservices architecture for applications running in containers, that doesn't mean it's the best choice for all of your applications, and your choice of architecture can greatly influence your choice of runtime. ***Ultimately, the goal is to have each application running in an environment that delivers the best return on investment for the business.***" [my emphasis]

Business automation should be driven by a form of lean value-stream – delivering value to the business without waste. What this means is, although some applications will convert easily and immediately to cloud (delivering mutable flexibility and moving from capex to opex expenditure, based on what you actually use), you should not neglect the modernisation of the rest of your business estate. You should exploit hybrid cloud to enable you to capitalise on your investment in your existing automation. There may be occasions (badly written applications tied into obsolete business practice, for example) where rewriting the automation from scratch for the Cloud makes sense; more often (we think) modernisation for hybrid cloud will be more cost-effective overall and less disruptive for the business. In other words, it will deliver more business value with less waste.

"What this means is, although some applications will convert easily and immediately to cloud... you should not neglect the modernisation of the rest of your business estate."

About the author

DAVID NORFOLK
Practice Leader:
Development & Governance

David Norfolk was working in the Research School of Chemistry at the Australian National University in the 1970s, when he discovered that computers could deliver misleading answers, even when programmed by very clever people. His ongoing interest in getting computers to deliver useful automation culminated in his joining Bloor in 2007 and taking on the development brief.

Development here refers to developing automated business outcomes, not just coding. It also covers the processes behind automation and the people issues associated with implementing it. He sees organisational maturity as a prerequisite for implementing effective (measured) process automation and ITIL as a useful framework for automated service delivery. He also looks after Collaboration and Business Process Management for Bloor, and takes a lively interest in the reinvention of the Mainframe as an Enterprise Server.

David has an honours degree in Chemistry, a graduate qualification in Computing, and is a Chartered IT Professional. He has a somewhat rusty NetWare 5 CNE certification and is a Member of the British Computer Society (he is on the committee of its Configuration Management Specialist Group).

He has worked in database administration (DBA) and operations research for the Australian Public Service in Canberra. David then worked for Bank of America and Swiss Bank Corporation in the UK, holding positions in DBA, systems development method and standards, internal control, network management, technology risk and even PC support. He was instrumental in introducing a formal systems development process for the Bank of America Global Banking product in Croydon.

In 1992 he started a new career as a professional writer and analyst. He is a past co-editor/co-owner) of Application Development Advisor and was associate editor for the launch of Register Developer. He helped organise the first London CMMI Made Practical conference in 2005 and has written for most of the major computer industry publications.

He runs his own company, David Rhys Enterprises Ltd, from his home in Chippenham, where he also indulges a keen interest in photography (he holds a Royal Photographic Society ARPS distinction).

Bloor overview

Technology is enabling rapid business evolution. The opportunities are immense but if you do not adapt then you will not survive. So in the age of ***Mutable*** business Evolution is Essential to your success.

We'll show you the future and help you deliver it.

Bloor brings fresh technological thinking to help you navigate complex business situations, converting challenges into new opportunities for real growth, profitability and impact.

We provide actionable strategic insight through our innovative independent technology research, advisory and consulting services. We assist companies throughout their transformation journeys to stay relevant, bringing fresh thinking to complex business situations and turning challenges into new opportunities for real growth and profitability.

For over 25 years, Bloor has assisted companies to intelligently evolve: by embracing technology to adjust their strategies and achieve the best possible outcomes. At Bloor, we will help you challenge assumptions to consistently improve and succeed.

Copyright and disclaimer

Bloor Research International Ltd
20–22 Wenlock Road
LONDON N1 7GU
United Kingdom

tel: **+44 (0)1494 291 992**
web: **www.Bloorresearch.com**
email: **info@Bloorresearch.com**