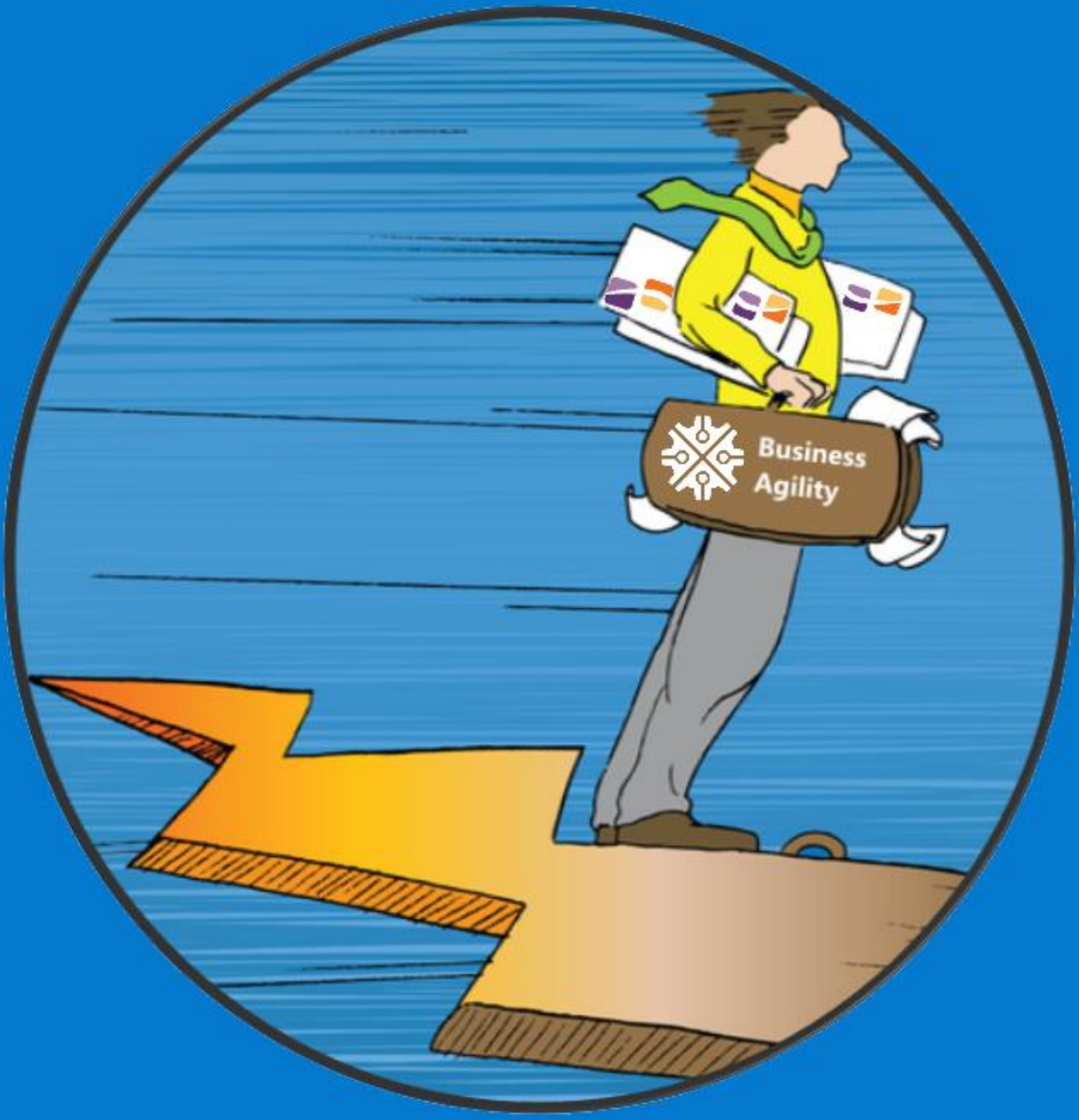
Adapting to Change Faster – Increasing Business Agility



Pliant Framework

✉ info@pliantframework.com
www.pliantframework.com
+61-399883994
P.O. Box 1024, Mitcham North, VIC 3132, Australia

Version 1.4

Introduction

Historically, product development in IT adapted many different methodologies to develop and deliver software. One of the main goals in this evolving process was to find a way to deliver software faster in order to receive feedback earlier in the process. Software development in IT started from traditional waterfall methodologies and moved to iterative and incremental methodologies and then agile and lean processes. Throughout all this, one of the main challenges for IT development teams was to deal with change and manage its impact. Changes include those in technology landscape, business needs, environment and rules and regulations. Despite all the efforts IT development teams put into improvement of their processes, business agility has not increased as expected.

What is Business Agility?

Gartner's report defined business agility as "an organization's ability to sense environmental change and respond efficiently and effectively to that change.[1]"

In other word, the ability to adapt to changing business requirements, decisions, policies and rules in a day-to-day business activity as fast as business people can determine the full business impact of the change, and assess whether change makes good business sense. This is an expectation that is hard to meet in IT and in particular the software development industry.

What is the challenge now?

In the software development industry, different modern methodologies and processes try to break down the big chunk of work into smaller measurable pieces that have a visible business value, and deliver those small pieces as soon as possible. This approach gives some agility to software development and release.

However from an IT perspective, some challenges related to the expected "business agility" still remain:

- No matter how small the task is, it requires, analysis, design, implementation and testing of that specific task.
- After the task is done, regression testing is required to make sure the overall quality is not affected.
- In addition, releasing has its own process, timeline and challenges.

The end result is that, these small pieces do not go out from IT to production and customers as fast as it is required by the business and demanded by customers. When a change is required in business logic, all the processes above are repeated,   no matter how fast that change is needed.

Business logic changes, may be needed for different reasons:

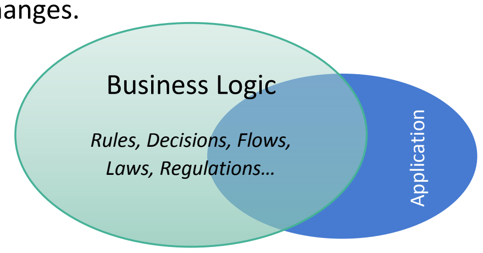
- Laws and regulation changes
- Expanding business and going after new target markets,

[1] https://www.gartner.com/doc/491436/defining-cultivating-measuring-enterprise-agility

- Adapting a behaviour to existing customers' needs
- Competition and market drivers
- Internationalization and adapting to countries' laws and regulations
- Testing and assessing customers' response to an uncertain decision …

These type of changes, are not due to a technology or engineering approach change, but they are a result of business decision (strategic, tactical and operational) changes which will end up changing the business logic of software.

In current solutions, dependencies between technology trend, software architecture and design constraints with business logic make it even harder to adapt to requiring business logic changes.



What is the current approach?

The current approach is Separation of Concerns (SoC), a design principal that separates software into distinct sections. Each section then addresses a separate concern which is information that affects the software in different ways. Each distinct area (which for example can be based on feature) has as little overlap in functionality as possible. Applying this design principal does not solve all issues, but reduces the impact of changes.

There are some fundamental issues that have not been addressed using this approach. ***THE ISSUES*** are

- The required changes usually need the commitment of the development (IT) team who is already occupied with their own backlog.
- When a change is made in the software code, regression testing is required which is expensive and time consuming.
- Releasing the change needs to be waited for most of the cases to get to the release vehicle.

In terms of resource and skill set allocation, time and costs effectiveness, this approach is not the best for the development team or the business.

Any other alternative?

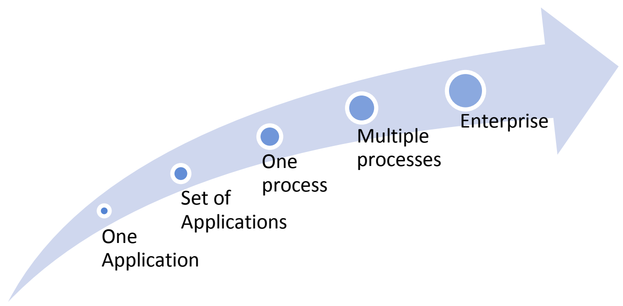
True business agility can be achieved by giving applications, systems and processes, the ability to change and adapt business rules and policies as part of their day-to-day business activities.

Hard coding your business rules will severely limit the degree of agility you can expect to achieve[2].

In this model, we still take advantage of Separation of Concerns but in a higher abstraction. In this abstraction, we allow business rules, decisions and policies to become the key driver of applications, systems and processes in a very flexible manner.

The goal of this SoC is to build an abstraction layer, ideally across the enterprise. This approach can be adapted incrementally and iteratively:

[2] http://lib-resources.unimelb.edu.au/gartner/research/138200/138218/138218.pdf

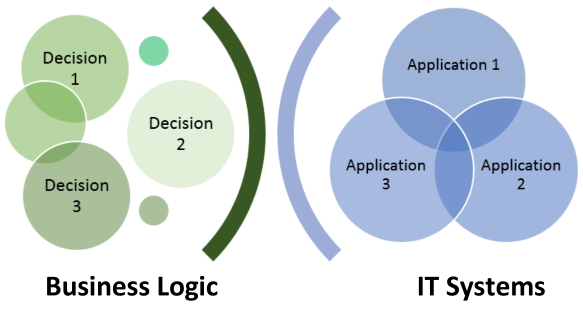


But to be clear, the main attribute of this abstraction layer is defined as follows:

It allows business rules, decisions and policies to become the first-class-citizens of enterprise, and drive behaviour of applications, systems and business processes in a flexible manner. This abstraction layer is maintainable not only by the IT and development team but also by less technical teams and business people with minimum involvement in IT (e.g. development and operational teams). **It also allows governance of all business logic across the whole lifecycle: from creating, testing and maintaining, to deploying, hosting, executing, monitoring and versioning.**

This **easy-to-manage business logic (business rules, decisions, and policies) platform** is the essential key to business agility which gives the control of business logic to business people.

A novel view of applications and business is now achieved:



As you can see this SoC clearly distinguishes between business logic of applications and the applications' implementation itself. Also the interaction between application and business logic are more isolated.

What are the benefits?

There are many benefits to this model:

- Business logic changes do not involve IT as much as traditional approaches. At the same time IT can be involved to the extent that they want.
- Changes occur in one abstraction that can be isolated, versioned, tested and deployed without changes to applications or deploying new applications at all!
- It provides a better tools for communication between IT and business people. This is due to the logic not being coded in the application but in less complex higher level executable models that both IT and business understand.
- Changes can be adapted, tested and released sooner. As the software code is not changed by the development team, changes can be isolated, versioned and deployed without changing the software applications hence eliminating a lengthy development process.
- Releases are streamlined, and can happen instantly.
- A centralized service-oriented platform for business logic (as a REST API) that can provide service not only to applications and products, but to external third parties as a business logic platform.

Business rules, decisions and policies are defining and driving business operations and behaviour.

Ultimate increase in business agility and productivity as business can response to a change in an easier, faster and cheaper way.

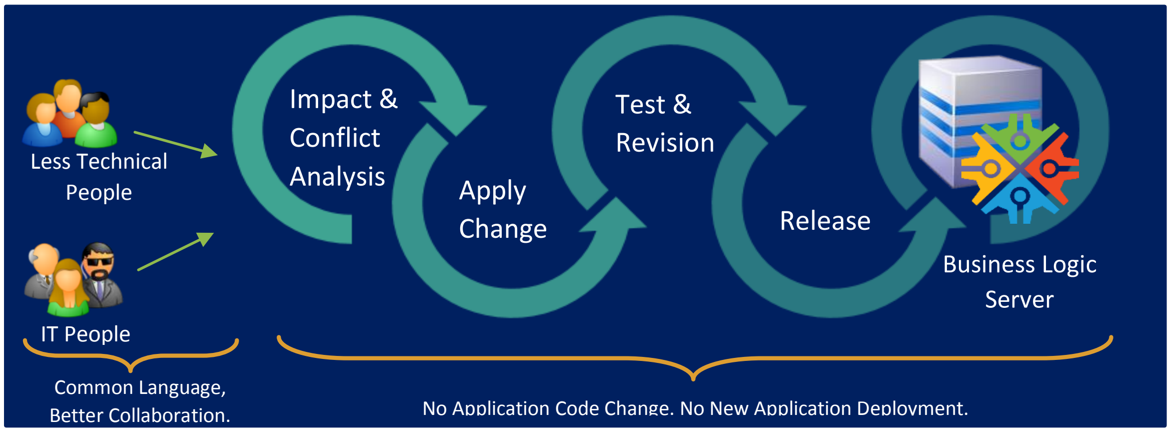
© Pliant Framework

info@pliantframework.com
www.pliantframework.com
+61-399883994
P.O. Box 1024, Mitcham North, VIC 3132, Australia

Business rule is a criterion used in business operations to guide behaviour, shape judgements and make decisions.[3]

True business agility can be achieved by making a business logic (business rules, decisions, policies…) platform as a single source of truth for business rules, decisions and policies that enables:

1. Better communication between business people (business analysts, subject matter experts…) and IT by establishing common vocabulary, language and modeling notation.
2. Impact analysis and understanding of business logic (rules, decisions and policies) conflicts of the change.
3. Changing business logic, without changing application code and without deploying new applications.
4. Business rules, decisions and policies can be served as a service and consumed throughout the enterprise by different applications, systems and processes.


Less Technical People
IT People
Common Language, Better Collaboration.
Impact & Conflict Analysis
Apply Change
Test & Revision
Release
Business Logic Server
No Application Code Change. No New Application Deployment.

[3] http://www.ronross.info/blog/2012/02/06/the-fundamental-problem-of-software-engineering-is-not-about-decisioning-2/

info@pliantframework.com
www.pliantframework.com
+61-399883994
P.O. Box 1024, Mitcham North, VIC 3132, Australia

For More Information Please Visit

http://www.flexrule.com

FlexRule

Flexible Software Made Easy

info@pliantframework.com
www.pliantframework.com
+61-399883994
P.O. Box 1024, Mitcham North, VIC 3132, Australia