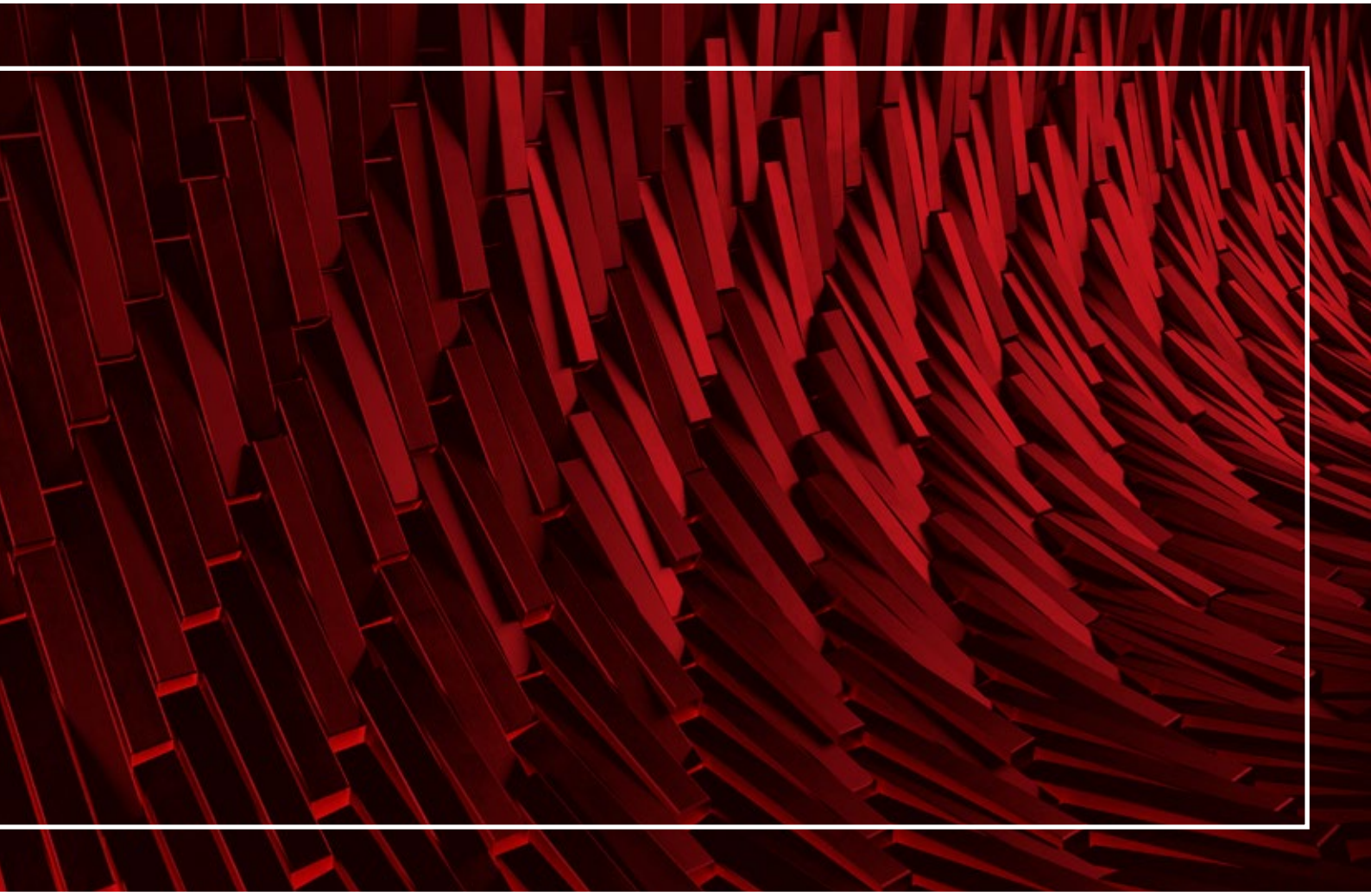
7 TIPS ON TRANSITIONING TO MICROSERVICES



CAPCO
THE FUTURE. NOW.

7 TIPS

ON TRANSITIONING TO MICROSERVICES

Microservices have been discussed and covered extensively ever since Martin Fowler started popularising the concept. Their benefits and drawbacks have been elegantly (and sometimes brutally) deconstructed on many occasions. In this article our intention is to provide a practical approach for a well-composed microservices implementation. Read our seven tips to consider before and during a transition from a monolith to a microservices architecture.

1. **"Does my organization really need microservices?"** This is a question that must be considered, as a microservices architecture is not suited to all enterprises. While its design can bring many benefits, it is complex to deliver and so organisations must be completely clear on their options. If you have gone through this thought process and the answer is still yes, perform an objective, self-aware review of your maturity, and your technical and business process fluency to apply a microservices architecture.

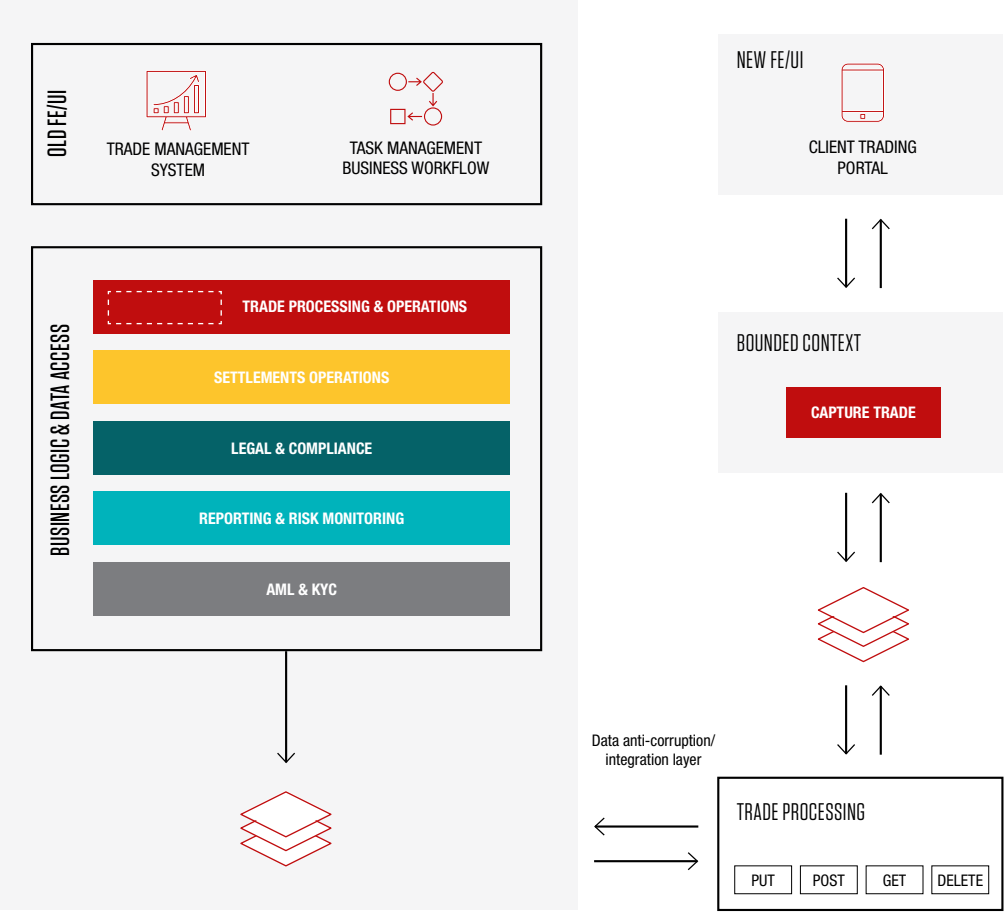
   Automation is crucial to a graceful, well-functioning microservices implementation. If your organisation is heavily attached to manual processing and intervention, make it a priority to address that before embarking on a microservices transition.

   Indirectly, the microservices approach encourages symmetry between a firm's architecture and the organisation of its people. This premise isn't new – it draws from Conway's Law (1968) and resonates with the mirroring hypothesis (Colfer and Baldwin, 2016). With that in mind, it's always necessary to perform an educational exercise across the firm, helping your employees understand the cultural and practical implications inherent to a microservices transformation, which is essentially building an architecture that reflects the way people work, interact and are organised.

2. Spend **a lot** of time delineating both your functional and wider, organisational frontiers (or bounded contexts). Ideally, a microservice will be a wrapper for a single piece of functionality and one independent team. Visualise the current construct of your business processes, their levels of interaction and other intrinsic dependencies. There shouldn't be similarities or excessive traffic between any two services – this should be immediately taken as a trigger to investigate a potential merger between them. Such verbose services will prove costly in the long-term, on various planes: for example, network performance issues and subtle, undesirable co-dependencies between business teams, causing delays and bottlenecks in developing, testing, and deploying changes and new features.

3. The monolith and any microservices should 'get used to' having each other around. Expect and prepare for a (long) period of co-existence and data-sharing between your legacy system and the new microservice platform, which should be evolved gradually. Select one of the least critical services for your pilot, experiment with it, record elements of failure and importantly, pay special attention to synchronising your data, by employing an anti-corruption layer – another fundamental Domain Driven Design concept. It is imperative to spend time designing this layer correctly so that it can be reused as part of future designs.

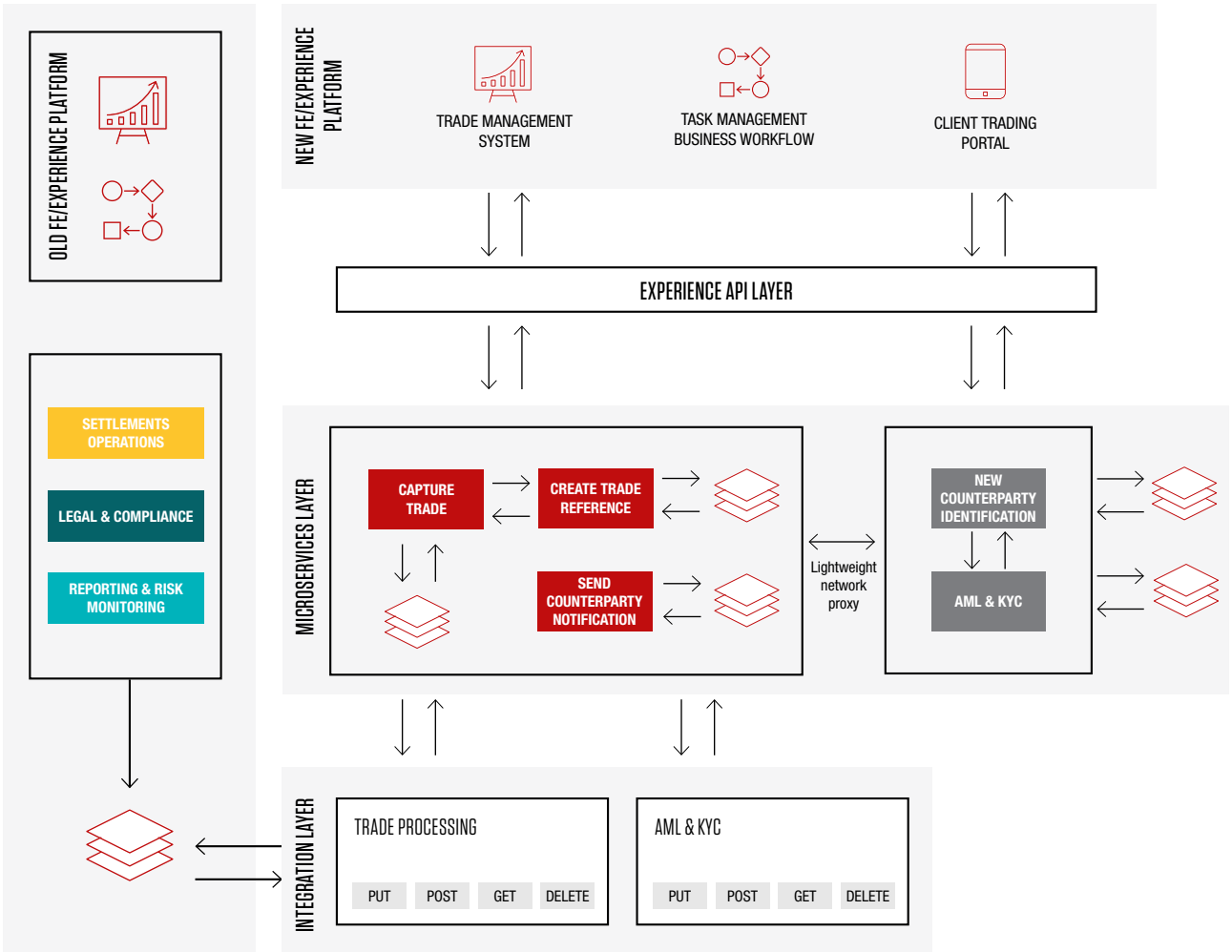
TRANSITION STATE: TACKLE THE CURRENT STATE IN BITE-SIZE PIECES



4. Any new solution should be designed with consideration of the operational aspects of the production implementation – and microservices architectures are no exception. As the volume of microservices in your enterprise grows, some early thought should be given to common non-functional elements that can be shared amongst a logical grouping of microservices. The precise list depends very much on several factors, but some common areas to think about would be intra-service network communications, security, deployment configuration, logging, and monitoring. Luckily, there are a growing number of sophisticated tools available to help but make sure this step is integral to the design, not part of a go-live checklist.

5. Of course, a true microservice should be self-contained, including reliance on physical data stores. Pick the data technology that is best-suited to your domain and let the anti-corruption layer handle the integration to any legacy data store.

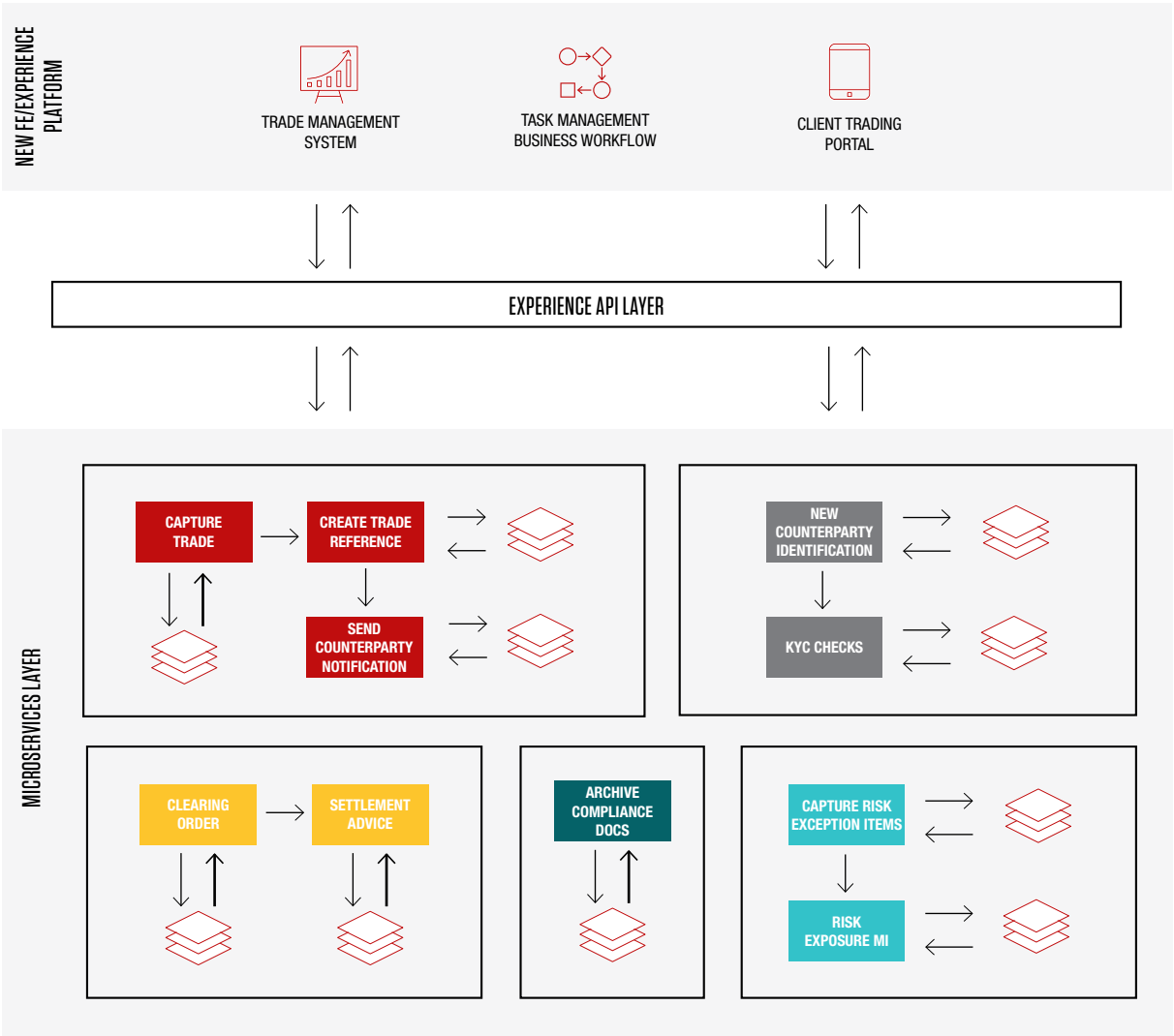
TRANSITION STATE: MOVING TO MICROSERVICES FAMILIES (SHARE COMMON CONFIGURATION PATTERNS)



6. One of the complexities of employing a microservice architecture is the shift in mindset from the reliance upon transactions to maintain a consistent state. It is quite acceptable for a business process to span across a number of microservices, so a design principle should be agreed on for the best option to do this when crossing a number of transactional boundaries.

7. Finally, don't be afraid to retire microservices as they become obsolete, nor to embrace new technologies that work for the microservice in question. Well-constructed architectures are the ones that adapt and evolve elegantly, without necessarily focussing on an end point.

TARGET STATE MICROSERVICES



FURTHER READING

Evans, E. (2003). Domain-Driven Design. Addison-Wesley, Pearson Education

Colfer, L. and Baldwin C. (2016). The Mirroring Hypothesis: Theory, Evidence and Exceptions. Harvard Business School Finance Working Paper No. 16-124

AUTHORS

Dave Cecil, Principal Consultant, Solution Architect
Beatrice Porcescu, Senior Consultant, Solution Architect

ABOUT CAPCO

Capco is a global technology and management consultancy dedicated to the financial services industry. Our professionals combine innovative thinking with unrivalled industry knowledge to offer our clients consulting expertise, complex technology and package integration, transformation delivery, and managed services, to move their organizations forward. Through our collaborative and efficient approach, we help our clients successfully innovate, increase revenue, manage risk and regulatory change, reduce costs, and enhance controls. We specialize primarily in banking, capital markets, wealth and investment management, and finance, risk & compliance. We also have an energy consulting practice. We serve our clients from offices in leading financial centers across the Americas, Europe, and Asia Pacific.

To learn more, visit our web site at **www.capco.com, or follow us on Twitter, Facebook, YouTube, LinkedIn and Xing.**

WORLDWIDE OFFICES

APAC
Bangalore
Bangkok
Hong Kong
Kuala Lumpur
Pune
Singapore

EUROPE
Bratislava
Brussels
Dusseldorf
Edinburgh
Frankfurt
Geneva
London
Paris
Stockholm
Vienna
Warsaw
Zurich

NORTH AMERICA
Charlotte
Chicago
Dallas
Houston
New York
Orlando
Toronto
Washington, DC

SOUTH AMERICA
São Paulo

WWW.CAPCO.COM

CAPCO
THE FUTURE. NOW.

© 2018 The Capital Markets Company NV. All rights reserved.