Sobreviver & Prosperar com Métodos Ágeis: Um Guia para Arquitetos Corporativos

Marc Lankhorst & Razvan Mitache

V2.0

Conteúdo

Introdução

O objetivo deste guia é *permitir que os arquitetos corporativos entreguem maior valor de negócio em ambientes ágeis*. Para atingir isso faremos duas coisas. Primeiro, demonstraremos que reunir a Arquitetura Corporativa com os Métodos Ágeis não somente é possível, como de fato é muito benéfico para ambas as áreas, e também para o negócio como um todo. Segundo, forneceremos conselhos sobre como integrar as duas disciplinas na prática, usando casos de uso reais e exemplos para dar aos praticantes as maiores possibilidades de sucesso. Em outras palavras, provaremos que isso pode ser feito e então mostraremos como fazê-lo. Sem mais demora, vamos a isso.

Parte 1: Arquitetura Corporativa e Métodos Ágeis se Misturam?

Em primeiro lugar, por que este guia é necessário? Vamos explorar este relacionamento aparentemente antagônico entre a Arquitetura Corporativa e os Métodos Ágeis.

Uma Falsa Disputa

Na maior parte da sua história a Arquitetura Corporativa tem sido uma atividade relativamente lenta e monolítica, que olhava longe no futuro e exaustivamente trabalhava para planejar as mudanças. Esta lentidão, aliado à falta de competências de comunicação dos arquitetos, e sua falha em adotar um senso de urgência quando lidando com as necessidades do negócio (a famosa Síndrome da Torre de Marfim), deixou a Arquitetura Corporativa com uma imagem muito desgastada. O fato de que por um longo tempo não havia ferramentas maduras para suportar os profissionais também não ajudou em nada. Naturalmente, as melhores práticas atuais de Arquitetura Corporativa se direcionam para longe desta passividade, mas práticas individuais de arquitetura fracamente gerenciadas, junto com o estigma do passado, significa que esta imagem é difícil de ser superada.

O movimento Ágil, por outro lado, abraça a velocidade e a responsividade, bem como uma abordagem de aprendizado pelo erro para resolver os problemas. Um negócio ágil é um negócio que foca na adaptabilidade para a mudança. Estes são os valores e princípios do Manifesto Ágil que sustentam este modelo mental, como uma referência:

Valores

- Indivíduos e interações mais do que processos e ferramentas.
- Software em funcionamento mais do que documentação abrangente.
- Colaboração com o cliente mais do que negociação de contratos.
- Responder a mudanças mais do que seguir um plano.

Princípios

1. Nossa maior prioridade é satisfazer o cliente por meio da entrega antecipada e contínua de software com valor.
2. Mudanças de requisitos são bem-vindas, mesmo em fases tardias do desenvolvimento. Os processos ágeis utilizam a mudança em favor da vantagem competitiva para o cliente.
3. Entregar software em funcionamento com frequência, desde algumas semanas até alguns meses, com uma preferência por prazos mais curtos.
4. As pessoas do negócio e os desenvolvedores devem trabalhar juntos diariamente ao longo do projeto.
5. Construa projetos em torno de indivíduos motivados. Dê-lhes o ambiente e o suporte que precisam e confie neles para realizarem o trabalho.
6. O método mais eficiente e efetivo de se transmitir informação para e dentro de uma equipe de desenvolvimento é a conversa face a face.
7. Software em funcionamento é a principal medida de progresso.
8. Os processos Ágeis promovem o desenvolvimento sustentável. Os patrocinadores, desenvolvedores e usuários devem ser capazes de manter um ritmo constante indefinidamente.
9. A atenção contínua à excelência técnica e a um bom projeto aumentam a agilidade.
10. Simplicidade – a arte de se maximizar a quantidade de trabalho não feito – é essencial.
11. As melhores arquiteturas, requisitos e projetos emergem de equipes que se auto organizam.
12. Em intervalos de tempo regulares, a equipe reflete sobre como se tornar mais efetiva, e então refina e ajusta seu comportamento de acordo.

Como evidenciado por isso, os Métodos Ágeis promovem a Colaboração, a Praticidade, a Responsividade, a Transparência, a Autonomia e a Confiança para garantir que a organização seja capaz de suportar a mudança. Estas características, em conjunto, produzem a *Agilidade*. Os Métodos Ágeis rejeitam o modelo em cascata de gerenciamento de projetos e aquilo que é conhecido como “o Grande Desenho Antecipado”, bem como uma documentação compreensiva e o planejamento de longo prazo.

Então, o problema para associar a Arquitetura Corporativa com os Métodos Ágeis tem origem em uma falsa percepção de incompatibilidade entre as duas práticas, que se originou nas deficiências históricas da Arquitetura Corporativa e na imagem negativa que elas criaram. Uma imagem de uma prática lenta e pesada, em completa antítese com o etos Ágil e, desta forma, presumivelmente incapaz de entregar qualquer valor em um ambiente Ágil. A Arquitetura Corporativa é percebida como lenta e “cautelosa”, enquanto o Ágil é rápido e “confiante”. Um é o passado; o outro é o futuro. Ou, pelo menos, é o que muitas pessoas parecem pensar enquanto elas deixam de notar sua complementariedade.

Nós, no entanto, pensamos diferente. Primeiro, nós reconhecemos que a Arquitetura Corporativa é, realmente, mais calculada e meticulosa por natureza, e, desta forma, ela se "move" em um ritmo mais lento. No entanto, isto acontece simplesmente porque ela é diferente dos Métodos Ágeis, não inferior. A Arquitetura Corporativa serve a um propósito diferente, usa meios e ferramentas diferentes, requer competências diferentes dos seus praticantes. Nós vamos falar mais sobre isso na medida em que formos avançando.

Velocidade Máxima, Sem Direção

Mudanças têm sido historicamente um grande problema para as empresas, e quanto maior a empresa mais dificuldades ela encontra para se adaptar às condições mutáveis do mercado. Uma grande parte da responsabilidade por isso advém da forma compreensiva como a mudança tem sido tratada – montes de documentação detalhada, iniciativas espalhadas por um longo horizonte de tempo, processos rígidos.

Enquanto isso, os métodos ágeis advogam dar pequenos passos com grande frequência para ser capaz de mudar a direção rapidamente, se necessário. A razão pela qual este modelo mental foi alçado ao topo e se tornou numa das tendências mais quentes do momento é que as abordagens ágeis funcionam realmente bem em um mundo cada vez mais volátil por causa das tecnologias revolucionárias, novas tendências sociais, regulação em constante mudança etc. Elas melhoram enormemente a responsividade às mudanças. Ainda assim, elas não são uma receita infalível para o sucesso.

Considere um ambiente somente ágil. Em organizações menores, algumas poucas equipes ágeis/DevOps podem coordenar a mudança entre elas mesmas, e as linhas de gerenciamento são curtas o suficiente para que a direção estratégica seja transmitida para as equipes diretamente. No entanto, quanto maior a empresa se torna, mais interconectados e interdependentes ficam os seus componentes (capacidades, recursos, processos, sistemas). Este panorama complexo de dependências é a razão pela qual a Arquitetura Corporativa é vital para alinhar as partes díspares da empresa com a direção estratégia geral, garantindo, assim, a adaptabilidade.

Empresas massivas podem ter centenas de equipes ágeis, cada uma trabalhando no seu próprio pequeno projeto, ignorando a grande figura, razão pela qual é necessário mais coordenação. Donos de produto (Product Owners) no nível das equipes geralmente não possuem esta percepção, o que torna provável que eles tomem decisões que podem ter consequências adversas no nível corporativo. De forma similar, se as equipes ágeis desconsideram o seu ambiente e terminam por construir silos ágeis, o resultado não será uma organização flexível. A falta de consciência – e, desta forma, de coordenação – ao final torna as futuras mudanças ainda mais difíceis. Quando aplicadas inadequadamente, as abordagens ágeis podem, na realidade, prejudicar a adaptabilidade geral da empresa.

Uma boa arquitetura é essencial neste caso. Naturalmente, os dias de modelar a empresa inteira na esperança de ser capaz de responder a qualquer pergunta estão bem longe no passado. Modelar todos os estados atuais e futuros, juntamente com seus roteiros associados, e manter tudo isso atualizado manualmente - especialmente no paradigma de mudanças pequenas e constantes - é muito trabalhoso. No entanto, o fato inquestionável é que sem arquitetura você está apenas indo em alta velocidade para lugar nenhum. Como o Gato Que Ri afirma em Alice no País das Maravilhas, “Se você não sabe para onde está indo, qualquer estrada te levará até lá”.

Casos de uso para os Métodos Ágeis e a Arquitetura Corporativa

Aqui estão alguns casos de uso para combinar a arquitetura corporativa e os métodos ágeis. Eles ilustram como esta integração pode trazer valor para a organização.

Priorizar o Valor de Negócio

Com base em uma análise da sua arquitetura corporativa, você pode ver como os vários épicos e funcionalidades são relacionados com os processos de negócio, capacidades, metas de negócio e partes interessadas associadas. Rastreando através dos seus modelos da arquitetura, você pode encontrar para quais metas de negócio e partes interessadas uma funcionalidade contribui, e se esta funcionalidade é mais importante do que outras. Informação objetiva como esta pode ajudar os donos de produto (Product Owner) a colocar um fim na priorização "orientada pelo grito", onde os usuários com as vozes mais altas garantem que os seus desejos sejam atendidos.

Coordenar o Trabalho

Identificar dependências arquiteturais pode ser usado para planejar as atividades ágeis. Por exemplo, se uma funcionalidade A depende da funcionalidade B, construa B primeiro. Isto é ainda mais importante naquilo que o SAFe chama de "habilitadores” – aspectos da solução que não fornecem funcionalidade diretamente para os usuários, mas suportam esta funcionalidade indiretamente e permitem a evolução e qualidade no longo prazo. Seu conceito de "Pista Arquitetural" (*Architectural Runway*) é um exemplo disso, onde você constrói a fundação técnica das novas funcionalidades no momento apropriado. Este conceito pode, no entanto, ser estendido além do software apenas. Investir, digamos, nas competências da equipe, pode fornecer uma fundação para muitos desenvolvimentos e melhorias futuros.

Acompanhar o Progresso

Se alguma funcionalidade ou habilitador é atrasado, por exemplo, por causa de uma equipe ágil querendo empurrá-la para o próximo *sprint*, o impacto na solução total precisa ser fatorado naquela decisão. Existem várias boas razões para esta equipe priorizar alguma outra coisa no lugar, mas geralmente eles não têm informações suficientes para entender as consequências da sua decisão. Ser capaz de rastrear de volta para a meta geral demonstrará como uma decisão afetará um *release*, projeto, data de entrega etc. Isso também ajuda as equipes para fazer perguntas como: *Como isso impacta as metas de negócio da organização? Quais grupos de clientes serão afetados? Nós podemos mitigar estes efeitos, através da priorização de outras funcionalidades, possivelmente de outras equipes?*

É aqui que os modelos da arquitetura são de grande ajuda, especialmente para os habilitadores. Uma vez que os habilitadores não fornecem funcionalidades para o usuário final, poucas pessoas reclamam quando eles são adiados. No entanto, eles são em geral muito importantes para várias funcionalidades diferentes e para a viabilidade da solução no longo prazo, de forma que manter a rastreabilidade destas dependências é muito importante.

Atualizar a Arquitetura

Em um contexto ágil, uma arquitetura corporativa não é apenas uma descrição estática de um estado futuro de longo prazo. Ao invés disso, ela é um instrumento compartilhado para fornecer a visão, dar a direção e garantir a coerência através da empresa, com base tanto em entradas de cima para baixo como de baixo para cima. Estratégia, metas e resultados desejados do negócio fornecem a direção de cima para baixo, enquanto as inovações e melhorias locais oferecem as mudanças de baixo para cima. Assim, os vários grupos na empresa trabalham juntos na arquitetura a partir de suas próprias perspectivas.

Casos de Uso para DevOps e Arquitetura Corporativa

Tendo visto algumas aplicações valiosas da arquitetura corporativa dentro do domínio dos métodos ágeis, vamos considerar, agora, como a arquitetura corporativa se relaciona com DevOps. Aqui estão alguns casos de uso principais.

Correlacionar & Propagar Eventos

Dar sentido a grandes panoramas organizacionais é difícil, ou praticamente impossível, sem a ajuda da arquitetura. Vamos dizer que você precisa explorar como eventos de segurança, ou seja, engenharia social, *phishing*, vírus e intrusão física, se propagam através da arquitetura para preparar uma avaliação e recomendações de risco sensatos.

Como você poderia saber o que está relacionado com o que, sem uma visão mais ampla da sua organização e de seus componentes? Você certamente poderia implementar algumas medidas de segurança apressadamente, e dizer para a Diretoria que você está tomando sólidas medidas para proteger a organização, mas isso seria o equivalente a manejar uma espada no escuro. Você pode manter o inimigo à distância, ou (mais provável) você pode não estar conseguindo absolutamente nada. DevOps pode, com certeza, se beneficiar das entradas da arquitetura.

Encontrar a Causa de um Problema

Identificar problemas como parte das iniciativas de mudança é significativamente mais fácil quando lidando com representações mais abstratas. A Arquitetura Corporativa não é nada mais do que um bom exemplo de uma representação deste tipo. Assim sendo, ela suporta métodos e técnicas valiosas para elicitar os problemas existentes ou potenciais, graças a coisas como a análise da causa-raiz. Esta abordagem funciona bem mesmo quando lidando com uma complexa rede de sistemas.

Continuidade do Negócio

Finalmente, a Arquitetura Corporativa pode ajudar o DevOps a garantir a continuidade do negócio. Isso é porque ela ajuda a responder a questões tais como: *Qual seria o impacto potencial de uma mudança?* Ou *O que pode dar errado se nós mudarmos este sistema, e como nós podemos tomar medidas mitigadoras em relação a isso?* Por realçar as principais dependências entre os vários aspectos da empresa (tanto dentro de um domínio, mas também entre domínios, por exemplo, como um elemento de infraestrutura tecnológica suporta uma capacidade importante do negócio) e permitir análises de cenário e de impacto, a arquitetura é uma fonte confiável de informação valiosa para os departamentos de DevOps.

Parte 2: Como Eu Integro a Arquitetura Corporativa e os Métodos Ágeis?

A Parte 1 desenvolveu a ideia de que a Arquitetura Corporativa e os Métodos Ágeis são compatíveis, e que o negócio pode se beneficiar enormemente ao fornecer uma fundação de Arquitetura Corporativa para suas práticas Ágeis. Agora, vamos discorrer sobre qual é o papel dos arquitetos corporativos em um contexto ágil, e também ver como a integração entre a Arquitetura Corporativa e os Métodos Ágeis pode se parecer na prática, usando modelos da arquitetura e o suporte adequado de ferramentas.

Preenchendo a Lacuna

Abordagens como o Scaled Agile Framework (SAFe), Disciplined Agile Delivery (DAD) e Large-Scale Scrum (LeSS) foram desenvolvidas exatamente para permitir a coordenação entre as equipes no contexto de grandes organizações. Ainda assim, nós consideramos suas abordagens carentes de arquitetura. Embora elas deem um passo na direção certa, elas são também muito focadas nas necessidades do desenvolvimento de software, o que torna suas perspectivas pesadamente enviesadas em direção a soluções de *software*. Felizmente, há mais em uma empresa do que *software*.

Tomemos o SAFe, por exemplo (veja a Figura 1). Ele usa uma abordagem iterativa em camadas, onde encontramos as equipes ágeis na camada mais baixa. Elas entregam seus resultados em uma frequência ágil típica de 2 a 3 semanas. No nível imediatamente acima os resultados destas equipes são integrados e liberados usando conceitos de arquitetura de soluções, como Pista Arquitetural (*Architectural Runway*) e Trem de Entrega Ágil (*ART-Agile Release Train*), o que garante que isso tudo se encaixe. Esta camada de Programas itera em uma velocidade que é algumas vezes a velocidade do nível de Equipes, de forma que ela forneça produtos entregáveis a cada 2 ou 3 meses.

O nível de Grandes Soluções cuida das soluções grandes e complexas, abrangendo vários Trens de Entrega Ágil e fornecedores, endereça a conformidade regulatória, padrões e outras restrições externas, e fornece limites financeiros para orientar a tomada de decisões.

Finalmente, o nível de Portfólio, no topo, é onde os desenvolvimentos grandes e de longo prazo estão posicionados. É também aqui onde a Arquitetura Corporativa pode, finalmente, encontrar um ponto de entrada e achar o seu próprio lugar. A estratégia de negócio alimenta esta camada e fornece o contexto para decisões de arquitetura em larga escala e de alto impacto, definição de prioridades e alocação de orçamento.

Este nível superior é onde os arquitetos corporativos poderiam avaliar a utilização de metodologias como o TOGAF para buscar e criar o valor. O TOGAF também tem uma estrutura iterativa, mostrada pelo familiar diagrama de círculos de colheita do seu Método de Desenvolvimento da Arquitetura (*ADM-Architecture Development Method*). No entanto, as iterações no ADM não fazem dele uma abordagem de arquitetura ágil, uma vez que a agilidade verdadeira necessita de iterações ponta-a-ponta desde a ideia até a solução pronta, e de volta.

Figura 1. A abordagem em camadas, iterativa, do Scaled Agile Framework

Aplicar o TOGAF em um contexto ágil precisa, desta forma, de uma adaptação considerável. Em primeiro lugar, os arquitetos precisarão se tornar mais voltados para fora e, por conseguinte, mais orientados para o negócio, ao construir e gerenciar sua arquitetura, por exemplo, prestando atenção nos clientes finais e nos resultados de negócio desejados. Um resultado de negócio poderia ser um produto para cliente final, que é descrito por meio de serviços, capacidades, produtos de software e outros artefatos, mas também pode ser uma transformação de negócio que implementa uma nova estratégia. Apenas, lembre-se de se ater ao tipo de arquitetura "apenas oportuna, apenas suficiente" que estamos defendendo.

Mais ainda, ao invés de primeiro fazer uma ou mais iterações da arquitetura no ADM e só então endereçar a realização daquela arquitetura e 'governar' a implementação na Fase G, as fases do ADM deveriam ser consideradas como correndo em paralelo, onde cada uma fornece resultados significativos em uma cadência regular, exatamente como outras atividades em um contexto ágil.

A Figura 2 dá uma ideia sobre isso. No topo vemos o desenvolvimento e a evolução da visão da empresa. Em seguida, as arquiteturas de linha de base são atualizadas continuamente com informações sobre o estado atual, fornecidas, por exemplo, pelas equipes ágeis e de operações/DevOps. Em uma situação ideal, esta informação é obtida diretamente de ferramentas como CMDBs, que monitoram o ambiente operacional, ou de ferramentas de mineração de processos, por exemplo.

O próximo nível abaixo mostra a evolução das arquiteturas alvo. É aqui onde os arquitetos corporativos e de domínio aplicam a maior parte do seu esforço. Finalmente, na parte de baixo, os arquitetos orientam e governam a mudança em interação contínua com as equipes ágeis. Isso não é apenas uma via de mão única, onde os arquitetos fornecem a direção para os desenvolvedores.

Fechar o ciclo de retroalimentação é essencial para fornecer o retorno do cliente, injetar ideias inovadoras sobre novas tecnologias e, em geral, ajustar as arquiteturas e priorizar os desenvolvimentos sob circunstâncias mutáveis para otimizar o valor de negócio criado. Este ciclo de retroalimentação curto é um fator crítico de sucesso para as abordagens ágeis, e os arquitetos deveriam aprender com isso.

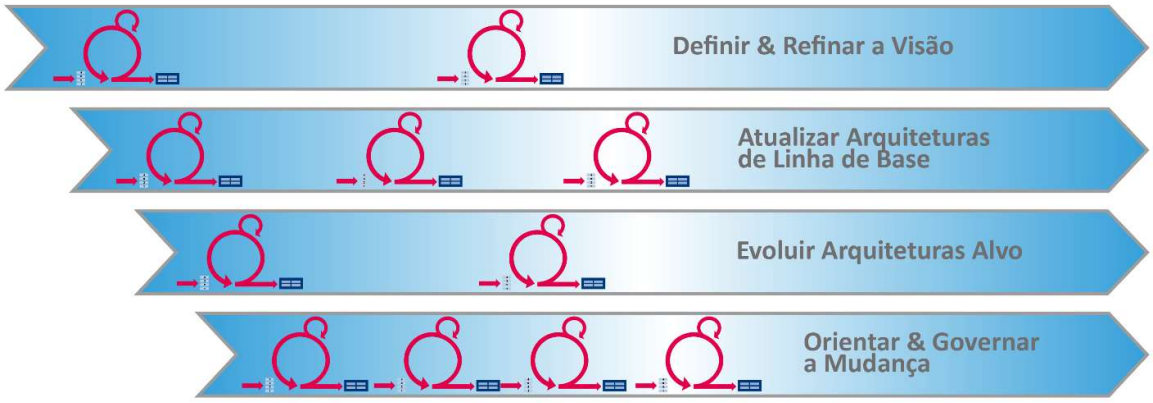

Figura 2. Transformando o TOGAF ADM em uma prática de arquitetura orientada para fluxos de valor

A abordagem do TOGAF, com foco pesado na documentação, também não se encaixa bem em um contexto ágil. Ao invés de documentos, as várias atividades de arquitetura deveriam fornecer um fluxo contínuo de valor na forma de *decisões* da arquitetura. Estas decisões servem tanto como entradas para outras fases como para orientar a implementação. Algumas decisões podem ser transmitidas pelos arquitetos diretamente para as equipes ágeis, outras precisarão ser registradas de uma forma que as torne acessíveis para todos os envolvidos. Ao invés de documentação extensiva, que é difícil de pesquisar ou analisar, acreditamos que modelos da arquitetura são uma forma muito melhor para registrar estas decisões.

Finalmente, o principal valor que a arquitetura trás para a mesa é que ela pode aumentar a agilidade das soluções que você está desenvolvendo. Ou seja, você precisa garantir não apenas a agilidade do processo de desenvolvimento, mas também a agilidade do resultado final que você está construindo (dentro do contexto organizacional).

Afinal, o que se segue depois que o processo de desenvolvimento do *software* termina é o mais importante, uma vez que é como uma solução passará a maior parte do seu ciclo de vida – como um produto acabado (esperamos) criando valor para o negócio. Não faz muito sentido desenvolver um *software* fantástico da forma correta, se ele é depois lançado para utilização em uma organização lenta, extremamente complexa e ineficiente. Ao final, ele não fará nenhuma diferença – a empresa se move na velocidade da sua parte mais lenta.

Mas o desenvolvimento ágil de *software* foca em responder rapidamente a requisitos em constante mudança; ele não endereça realmente a agilidade dos produtos resultantes. Com frequência, há uma pressuposição subjacente de que *software* é ágil 'por definição', uma vez que você sempre poderá modificá-lo, por exemplo, através de refatoração. Embora *software* possa ser mais ágil do que os construtos no mundo físico, este pressuposto é, no mínimo, ingênuo, e potencialmente perigoso.

Como Grady Booch uma vez definiu, a arquitetura considera aquelas decisões de desenho que são caras para mudar. Por exemplo, há alguns anos a BiZZdesign mudou sua plataforma de arquitetura de um modelo centralizado para um modelo distribuído. Esta é a típica decisão arquitetural que tem um impacto profundo e não pode ser tomada de forma inconsequente, uma vez que demandaria um esforço substancial mudar isso novamente. Uma prática sólida da arquitetura ajuda você a identificar tais decisões, evitar erros e promover soluções que são fáceis de adaptar nas áreas onde se pode esperar mudanças. Vamos expandir este assunto.

Agilidade Sistêmica é Fundamental para a Mistura

Chamamos este outro tipo de agilidade de Agilidade Sistêmica. Definimos Agilidade Sistêmica como ter sistemas técnicos e organizacionais que sejam fáceis de reconfigurar, adaptar e estender quando surge a necessidade. Um sistema ágil é um que simplifica ações como:

Fazer mudanças

Quão fácil é fazer alguma mudança no sistema técnico e/ou organizacional da empresa? Alguns aspectos relevantes são personabilidade (pelos usuários), adaptabilidade (por gerenciamento de sistemas), analisabilidade (pelos desenhistas), mutabilidade (pelos desenvolvedores), escalabilidade (por exemplo, para acomodar grandes volumes). Para garantir que estas funcionalidades sejam incrementadas efetivamente, existem várias boas práticas que os arquitetos deveriam seguir.

As mais importantes destas boas práticas são:

- suportar a separação de preocupações (por exemplo, usar estratificação e modularização para concentrar funcionalidade em lugares específicos, de modo a tornar as mudanças tão localizadas quanto possível)

- garantir o baixo acoplamento e alta coesão (isso é, um baixo número de relacionamentos entre sistemas e uma alta coesão interna, o que simplifica a análise e evita que as mudanças sejam propagadas através do sistema inteiro)
- e, finalmente, encapsulamento (isso é, criar interfaces claras e não permitir o acesso desde o ambiente para a implementação interna, cuja grande força é permitir que a implementação seja modificada sem afetar o ambiente)

Entregar mudanças

Isso é particularmente importante no contexto dos processos ágil e DevOps, onde as mudanças são implementadas com frequência, e a retroalimentação é coletada mais cedo. Os aspectos principais nos quais os arquitetos devem manter a atenção são:

- Facilidade de aprendizado: Quão fácil é para os vários grupos de partes interessadas aprender como trabalhar com os novos sistemas, procedimentos etc.?
- Facilidade de instalação: Quão fácil é implementar um novo sistema?
- Facilidade de teste: Quão fácil é testar o novo sistema, particularmente quando comparado com outros sistemas existentes?
- Facilidade de gerenciamento: Quão fácil é gerenciar, controlar e manter o sistema?

Levar estas variáveis em consideração e torná-las parte do seu sistema de avaliação certamente melhorará a sua capacidade de implementação de novos sistemas que suportam a agilidade, ao invés de diminuí-la.

Lidar com as consequências das mudanças

Isto significa simplesmente que se alguma coisa der errado durante uma mudança, os efeitos deveriam ser mínimos e corrigidos facilmente para minimizar o risco de interrupção nas operações do dia-a-dia. Tolerância a falhas, redundância e mecanismos simplificados de recuperação facilitam isso.

Integrar soluções

Digamos que um processo de negócio, sistema ou componente de informação, ou outro elemento, novo ou alterado. precisa ser conectado com o panorama existente. Isto requer fácil interoperabilidade. Como a Arquitetura Corporativa pode deixar a sua marca aqui? A resposta é: por meio da evangelização e imposição de padrões relevantes por toda a empresa. Nós voltaremos a este tema logo abaixo, mas por enquanto vamos em frente.

Desacoplar soluções

A facilidade de desacoplamento é importante, porque um sistema ágil deveria evitar que mudanças localizadas se propagassem de forma desordenada. Além disso, a reusabilidade é incentivada se os elementos podem ser facilmente tirados do seu contexto e conectados em outro lugar, ou seja, não existem dependências desnecessárias em relação ao seu ambiente. A organização deveria objetivar o desenho de componentes independentes e reusáveis, o que requer uma retaguarda de arquitetura que forneça o contexto para o reuso – assim, a arquitetura corporativa entra em cena.

Então, qual é o maior inimigo da agilidade sistêmica? Colocando de forma direta, a *complexidade* – ter vários relacionamentos entre os elementos de um sistema. Isso acontece porque a complexidade requer um longo tempo para ser compreendida (o que é necessário para uma tomada de decisões informada), as mudanças em si são muito mais trabalhosas, mudanças localizadas podem ter muitos efeitos colaterais não previstos (algumas vezes, em lugares bastante inesperados) e, finalmente, porque ela força você a realizar muitos testes. Para lidar com os desafios da complexidade, nós recomendamos duas práticas – particionamento e alinhamento.

Particionamento é a estratégia principal para controlar a complexidade. Experimente criar subdivisões autônomas da sua arquitetura e definir responsabilidades claras sobre cada uma delas. Enquanto você faz isso, concentre-se nos relacionamentos sinérgicos onde 'Se A precisa de B *e* B precisa de A, então eles pertencem à mesma partição, porque eles sempre serão usados juntos'. No entanto, se este não é o caso, eles são autônomos, e você pode colocá-los em partições separadas. Trabalhe para atingir um balanço entre controlar a complexidade *dentro* dos sistemas e a necessidade de interoperabilidade *entre* os sistemas.

Particionar também simplifica o alinhamento de negócio, uma vez que ele também reduz a complexidade do processo de decisão, isso é, ter muitas pessoas/unidades organizacionais que têm alguma coisa a dizer sobre o sistema. Um processo de decisão complicado é geralmente resultado da falta de alinhamento entre as capacidades de negócio e os sistemas de TI subjacentes. Por exemplo, imagine ter um sistema monolítico que precisa servir tanto ao departamento de atendimento comercial (focado na criação rápida de novas ofertas de produtos) bem como à retaguarda administrativa (focada na estabilidade).

Então, para garantir o alinhamento, os arquitetos deveriam objetivar a criação de capacidades de negócio autônomas (usando seu conhecimento abrangente da empresa) que são responsáveis por seus próprios processos, sistemas e dados.

Padronização

Agora, vamos tomar um momento para endereçar o assunto da padronização, o qual mencionamos brevemente antes, uma vez que mencionar isso em um guia ágil deve ter suscitado alguns questionamentos, com certeza. O uso de padrões é um dos meios pelos quais a arquitetura corporativa garante que as várias partes do quebra-cabeças (organizacional) se encaixam perfeitamente, entregando, assim, alinhamento e eficiência.

No caso do desenvolvimento e integração ágil de soluções, a clareza e a estrutura possibilitados pela padronização não podem ser desprezados. Naturalmente, à primeira vista a padronização e a agilidade parecem ser bastante contraditórios. Afinal, padrões são coisas rígidas, correto? Bem, a resposta é basicamente sim, eles são inflexíveis, mas – e esta é a pegadinha – isto não é realmente algo ruim para o nosso propósito.

Quando aplicado de forma correta, o uso de padrões aumenta a flexibilidade e a agilidade, porque ao especificar as 'regras do jogo', isso facilita o funcionamento dentro de um 'campo de jogo' bem definido. No caso de *software*, isso significa desenvolver soluções com interfaces padronizadas que combinam e podem ser reusadas com pouco esforço. Pense nos pinos das tomadas elétricas ou nos tijolos de Lego, por exemplo. Então, se há uma padronização 'boa', mas também há uma padronização 'ruim', como você pode ter a certeza de que está fazendo do jeito certo?

Bem, não levando em consideração em que indústria sua empresa opera, ou as mudanças específicas que você está tentando fazer, dois tipos de padronização se destacam, chamadas longitudinal (boa) e transversal (ruim). Vamos dar uma olhada em ambas.

Padronização Longitudinal

No caso da padronização longitudinal, os mesmos passos em um processo de negócio têm que ser implementados para várias linhas de produto, serviços ou segmentos de clientes, como você pode ver na Figura 3. Isso facilita muito o reuso de um passo do processo para produtos/serviços/segmentos diferentes, o que melhora sua agilidade, uma vez que o reuso suporta uma velocidade maior de mudança.

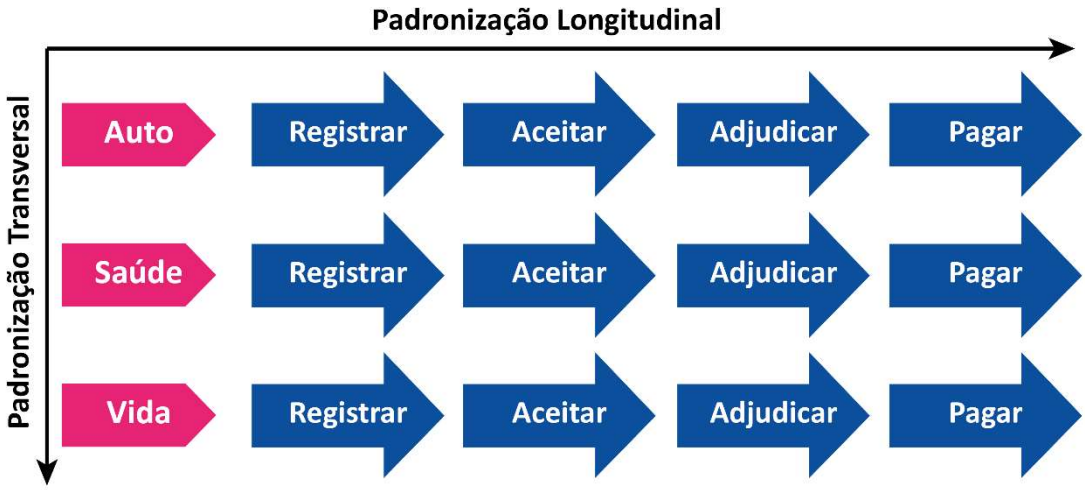

Figura 3. Padronização boa

Padronização Transversal

A padronização transversal, por outro lado, se refere a usar a mesma implementação dentro de um passo para todos os produtos, serviços ou segmentos (veja a Figura 4). Isso provavelmente prejudicará sua agilidade, porque o resultado pode se tornar muito complexo e, desta forma, complicado de mudar. Quando processos de diferentes produtos, serviços ou segmentos se tornam interdependentes, o resultado final é que a mudança se propaga através da organização ou do sistema, o teste se torna mais difícil, e a agilidade sofre.

Figura 4. Padronização ruim

Os arquitetos corporativos deveriam, assim, desenhar unidades separadas que podem ser modificadas de forma independente. Quanto menor a unidade de mudança, mais fácil executar aquela mudança. A arquitetura deveria facilitar ter blocos de construção que tenham baixa complexidade (isso é, tenham poucas dependências internas), pouca ou nenhuma interdependência com outros blocos de construção, e que sejam fáceis de testar localmente.

Adicionalmente, para suportar a criação dos blocos de construção corretos, você deveria idealmente conter a propagação das mudanças dentro das unidades organizacionais (departamentos, equipes etc.) tanto quanto possível. Isso significa concentrar o conhecimento necessário para as mudanças e sempre atribuir responsabilidades claras.

O Papel das Plataformas & Modelos na Agilidade Sistêmica

Uma vez que nós exploramos o assunto da agilidade sistêmica, também gostaríamos de mencionar as plataformas de negócio, que representam uma importante base para a agilidade sistêmica. Então, o que é infraestrutura de negócio? Colocando de forma simples, infraestrutura de negócio representa os sistemas e capacidades que quase nunca mudam, e como tal podem servir como uma fundação, ou infraestrutura, para as coisas que tendem a mudar com (bastante) mais frequência.

Para lhe dar uma ideia disso, considere a fiação elétrica da sua casa ou escritório. Isso quase nunca muda durante toda a vida útil da sua casa ou escritório. O que você conecta nela, no entanto, varia dramaticamente. Apenas pense nos seus aparelhos elétricos nos anos 1980, e compare-os com o que você possui hoje. O mesmo se aplica para os sistemas de informação da sua empresa.

Então, por exemplo, se a sua estrutura de dados é estável, mas os dados em si são mutáveis, faz muito sentido usar um SGDB. No entanto, se a sua estrutura de dados muda constantemente, então você provavelmente viverá melhor sem ele. Isso é completamente aplicável em um contexto de negócio. Pense a respeito de suas regras de negócio – se elas tendem a mudar com frequência, é realmente desejável que elas estejam codificadas rigidamente em procedimentos armazenados (*stored procedures*) no seu banco de dados, ou nos seus programas? Provavelmente não.

Naturalmente, você precisa determinar cuidadosamente onde, exatamente, você deseja ser ágil. Isso porque, não se engane, ser extremamente flexível em todos os aspectos não é nem necessário, nem realístico, e é também muito caro. Uma vez que as coisas na sua organização e nos seus sistemas não mudam todas no mesmo ritmo, você precisa avaliar a situação e decidir de acordo com os tipos de mudança que você precisa.

Nós acreditamos que a melhor forma de fazer isso é alavancando os modelos de arquitetura para realizar análises de dependência, impacto, valor técnico e de negócio, e outros tipos de análises, de forma que você tenha um entendimento claro do impacto da mudança.

Confiar em modelos é consistente com a tendência geral da TI em direção a níveis mais altos de abstração. Isso porque mudanças nos níveis mais altos de abstração são mais fáceis, mais rápidas, e com menor propensão a erros. Então, automatizar as etapas de implementação tanto quanto possível provavelmente contribuirá para a agilidade da sua organização, como mostrado na Figura 5. Exemplos disso incluem especificações executáveis, tais como modelos de processo BPMN, modelos de decisão DMN, transformações automatizadas de modelos, geração de código a partir de modelos, teste e geração de casos de teste automatizados a partir de modelos, e implementação automatizada.

Se você precisa de agilidade no seu conhecimento de negócio (processos, regras, dados), sua infraestrutura de negócio deveria suportar isso. O que você precisa, neste caso, é um ambiente de execução no qual você pode rodar todo o seu conhecimento de negócio. Através do uso de modelos para identificar percepções valiosas, ao invés de ter que se apoiar no próprio código, você pode obter benefícios significativos. Isso inclui a necessidade de testes menos compreensivos das mudanças no software, o envolvimento de menos tipos de especialistas, e menos erros na tradução das especificações orientadas para o negócio (por exemplo, regras de negócio, legislação) para a implementação. Isso é ilustrado na Figura 5.

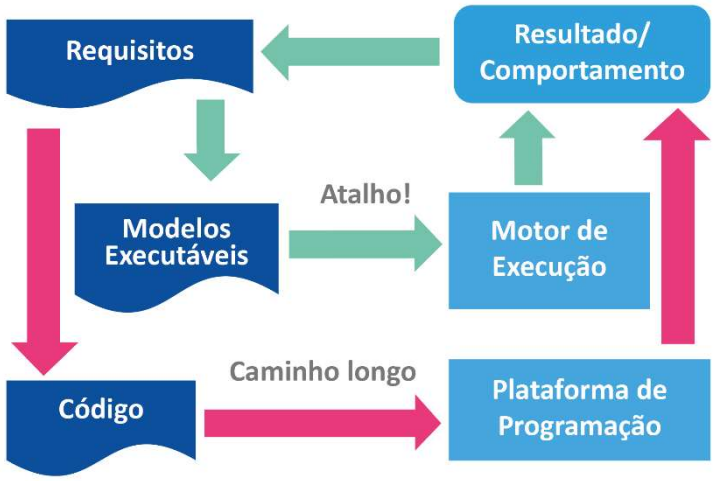

Figura 5. Agilidade por meio de modelos

Além do mais, ter uma plataforma como essa poderia também se encaixar com a iteração rápida dos processos ágeis. Aqui estão alguns poucos exemplos de como uma plataforma de negócio configurável poderia se parecer:

- Gerenciamento de processos de negócio e sistemas de gerenciamento de fluxo de trabalho
- Motores de regras de negócio
- Plataformas de desenvolvimento de pouca codificação
- Formulários Web configuráveis e inteligentes
- Configuração gráfica dos modelos de dados para os bancos de dados

Visão Ampla = Melhores Decisões = Maior Agilidade

Com sorte, isso pode lhe dar alguma perspectiva sobre a verdadeira agilidade dos negócios. Ao final, tudo isso pretende ajudar os arquitetos a entreabrir a porta e começar a pensar sobre as possibilidades. A conclusão é que todos os frameworks ágeis que mencionamos anteriormente – SAFe, DAD, LeSS – são bastante centrados na TI. Para ilustrar brevemente este ponto, o papel do arquiteto corporativo, de acordo com o SAFe, é "[...] orientar a implementação holística da tecnologia [....]". Nós acreditamos que um verdadeiro arquiteto corporativo não é focado apenas na tecnologia.

Mais ainda, muitas organizações já não desenvolvem seu próprio *software*, mas principalmente adquirem e integram pacotes de prateleira. As preocupações arquiteturais são concentradas, então, em quão bem estes pacotes suportam os requisitos relevantes de negócio hoje e em um futuro vislumbrável, e na sua integração com o restante do panorama de negócio e de TI. A arquitetura de negócio é, assim, uma parte cada vez mais importante desta equação.

Assim sendo, os praticantes individuais podem desempenhar um papel importante na habilitação do florescimento das práticas ágeis, trazendo para a mesa o mapeamento da estratégia, o planejamento baseado em capacidades, o mapeamento de valor, o gerenciamento dos processos de negócio, Lean-Six Sigma, e outras disciplinas relacionadas com o negócio. Elas oferecem uma oportunidade valiosa para melhorar a figura. Uma empresa verdadeiramente ágil precisa mais do que apenas uma TI ágil.

A arquitetura de solução pode muito bem ser cuidada pelas equipes ágeis, mas o seu trabalho deve se encaixar na visão geral da organização. Por exemplo, donos de produto estão focados principalmente nos requisitos funcionais das soluções individuais, mas quem toma conta dos requisitos não-funcionais, do uso de padrões, da conformidade regulatória, ou da direção estratégica? Quando você considera isso, os benefícios da visão ampla da Arquitetura Corporativa rapidamente se tornam claros.

A Arquitetura Corporativa é capaz de olhar para fora da arena do desenvolvimento de software e levar em consideração coisas como os resultados de negócio esperados, as capacidades a serem desenvolvidas ou melhoradas, os recursos necessários, os processos de negócio, a infraestrutura física e de TI a ser realizada, riscos de segurança, e outros aspectos relevantes. Assim sendo, os arquitetos corporativos podem adicionar valor ao traduzir a estratégia de negócio para um amplo espectro de decisões arquiteturais. Isso garante que um portfólio coerente de iniciativas seja mantido e previne que as equipes ou projetos construam seus próprios silos ágeis.

O Papel do Arquiteto Corporativo no Desenvolvimento Ágil

Um ambiente ágil pode ser uma experiência desafiadora para um profissional de arquitetura corporativa. Mesmo assim, como explicamos até aqui, existe um lugar para a arquitetura junto com os métodos ágeis. Aqui estão alguns pontos claros nos quais os arquitetos deveriam se concentrar para entregar valor para a organização:

- Traduzir a estratégia de negócio em iniciativas técnicas de alto nível sem ir para um nível detalhado, garantindo assim que a direção do desenvolvimento está em linha com as metas do negócio, enquanto se mantém longe do micro gerenciamento.
- Implementar padrões adequados, oferecendo interfaces padronizadas para as equipes ágeis e os habilitando para facilmente combinar e reusar blocos de construção.
- Suportar a gerência por meio do fornecimento de uma visão completa das iniciativas de desenvolvimento e solução de negócio.
- Transmitir esta visão para as equipes ágeis para garantir o alinhamento com a direção estratégica da empresa.
- Ajudar a definir a estratégia para construir e manter a pista arquitetural.
- Procurar elevar continuamente as práticas de modelagem, desenho e codificação.
- Estabelecer melhores práticas através da facilitação do reuso de código, componentes e padrões que tenham se provado bem-sucedidos.
- Organizar oficinas para encorajar a inovação.
- Gerar e analisar ideias inovadoras em torno de quais tecnologias deveriam ser empregadas pela empresa.
- Procurar a retroalimentação constante em relação às iniciativas em curso em toda a empresa, e realizar análises de custo-benefício quando for relevante.

Parte 3: Modelagem Ágil

Modelos são um instrumento importante para os arquitetos e desenvolvedores. Eles podem fornecer um importante contexto de negócio, ajudá-lo a analisar o impacto de uma mudança, comparar e avaliar soluções alternativas e muito mais. Assim sendo, gostaríamos de analisar o uso de modelos de arquitetura no contexto dos métodos ágeis.

Modelos ArchiMate e as Formas Ágeis de Trabalho

Para abordar o valor e utilidade de relacionar modelos de arquitetura com as formas ágeis de trabalho, nos concentramos especificamente no SAFe como um framework ágil representativo e na linguagem ArchiMate para arquitetura corporativa como a abordagem de modelagem. Por favor, consulte a figura simplificada anterior do SAFe (Figura 1). Ela mostra vários conceitos usados no SAFe, bem como em outros métodos ágeis – naturalmente, esta não é uma representação exaustiva do framework. Um dos aspectos proeminentes nesta estratificação é que quanto mais alto você sobe, mais o foco se move para a *intenção*, em vez da *construção* da solução.

Na Figura 6, mostramos como você pode usar elementos do ArchiMate para expressar estes conceitos. Por exemplo, no nível do *Portfólio*, os elementos de alto nível de motivação e estratégia do ArchiMate 3 (Curso de Ação, Capacidade, Recurso, Meta e Resultado) podem ser usados com muito sucesso. O planejamento baseado em capacidades oferece um bom ponto de partida para discussões de investimento neste nível e no seguinte.

SAFe	ArchiMate
Nível de Portfólio	
Tema Estratégico	Partes Interessadas, Motivadores, Metas, Curso de Ação
Épico (Negócio, Habilitador)	Resultado
Nível de Grande Solução	
Capacidades, Habilitadores	Capacidades, Recursos
Solução	Serviços de Negócio & Aplicativo, Produtos, Processos de Negócio
Framework Econômico	Princípios, Restrições
Nível de Programa	
Funcionalidades	Requisitos, Serviços de Aplicativo & Tecnologia
Pista Arquitetural	Restrições, Conceitos centrais
Incremento de Programa	Platô
Nível de Equipe	
Histórias de Usuário & da Arquitetura	Requisitos & Restrições, Funções de Aplicativo & Tecnologia
Incremento de Produto	Platô & Componentes de Aplicativo, Artefatos, Entregáveis

Figura 6. Exemplo mapeando elementos do ArchiMate para conceitos do SAFe

No nível de *Grandes Soluções*, a arquitetura de alto nível da solução é expressa em termos das Capacidades e Recursos, Produtos, Serviços de Negócio e de Aplicativo relevantes a serem desenvolvidos e os Processos de Negócio a serem suportados, e o framework econômico fornece Princípios orientadores e Restrições.

No nível de Programa, a direção é expressa por meio de Requisitos e Serviços desejados, o Roteiro por meio de vários Platôs, e a Pista Arquitetural é expressa por meio dos conceitos centrais do ArchiMate e restrições relevantes, tais como os padrões de tecnologia. Finalmente, no nível da Equipe, vemos as histórias de usuário e da arquitetura sendo expressas por meio dos Requisitos e Restrições, e as funções necessárias, enquanto o resultado do sprint (incremento do produto) consiste tipicamente de Componentes de Aplicativo, Artefatos de software e outros Entregáveis, o que resulta em um Platô (um estado estável da arquitetura).

Note que os detalhes da solução não são tipicamente descritos por meio de modelos ArchiMate. Ele é, no final das contas, uma linguagem destinada à Arquitetura Corporativa. O desenho detalhado é, em geral, melhor realizado usando UML, BPMN, ou outras linguagens de modelagem no nível da implementação. No entanto, é bastante fácil ligar seus modelos ArchiMate com este nível de implementação.

Por outro lado, vários conceitos na sua pista arquitetural poderiam ser mapeados diretamente para os elementos correspondente no nível de implementação, tais como Componentes, Interfaces e Serviços de Aplicativo, na UML, ou Processos de Negócio, no BPMN. A espinha arquitetural fornecida pelos modelos ArchiMate é inestimável no rastreamento das inúmeras dependências entre as várias iniciativas, programas e resultados, o que, como mencionado anteriormente, é um grande desafio para escalar os métodos ágeis para o nível corporativo.

Esta rastreabilidade entre os vários níveis significa que você sabe onde mudanças nas estratégias e requisitos podem ter um impacto, você pode assegurar a coerência e a estabilidade do seu panorama de TI, e você está comprovadamente no controle. Este último aspecto é cada vez mais importante em ambientes de negócio altamente regulados.

Arquitetura Intencional

Para ser mais claro, isso não é uma recomendação para criar grandes modelos ArchiMate para capturar o desenho completo antes que você comece a construir qualquer coisa. Os arquitetos deveriam objetivar criar e evoluir modelos oportunos que capturam a informação necessária para tomar a decisão correta no momento certo e demonstrar a intenção da arquitetura para as equipes de desenvolvimento.

Os conceitos do ArchiMate são uma ótima maneira de expressar essa arquitetura intencional. Isso ocorre porque o ArchiMate não foi desenvolvido para o desenho detalhado da solução. Seus conceitos têm como objetivo expressar os aspectos estruturais e comportamentais essenciais das soluções, e a motivação por trás delas. Além disso, o ArchiMate abrange toda a empresa e a TI e, assim, fornece às equipes ágeis contexto e clareza além das preocupações de desenvolvimento de software apenas. Por exemplo: Em quais processos de negócio isso será usado? Quem são os potenciais usuários e outras partes interessadas? Qual é a jornada do cliente que imaginamos? Como isso se relaciona com outros produtos em nosso portfólio? Como isso contribui para as principais metas de negócio? Para qualquer organização (mas especialmente em um contexto ágil), todos devem entender o objetivo de seus esforços.

Fornecer esse tipo de contexto e de intenção de nível mais alto é essencial para o desenvolvimento ágil. Em ambientes em cascata tradicionais, talvez você tenha tido analistas de negócio, analistas de informações e outros papéis com a tarefa específica para traduzir a intenção estratégica em requisitos concretos. No desenvolvimento ágil, no entanto, grande parte dessa responsabilidade recai sobre os ombros das equipes ágeis e dos donos de produto. Sem um claro entendimento do contexto, como você pode esperar que eles tomem as decisões certas sobre o projeto?

Modelos de Arquitetura para Suportar as Equipes Ágeis

Lidar com sistemas grandes e complexos é difícil em qualquer metodologia, e talvez seja o maior desafio na adoção de formas ágeis de trabalhar. Devido à sua própria natureza, esses grandes sistemas não são muito ágeis, pois o grande número de dependências que eles apresentam atrasa a mudança.

Alguns metodologistas ágeis diriam que você tem que 'simplesmente' dividir esses sistemas em blocos que sejam gerenciáveis por uma única equipe. Os microsserviços são um exemplo de tal abordagem, mas como qualquer pessoa que os tenha visto em ação sabe, o resultado é que a complexidade que costumava ficar escondida dentro de um único grande sistema agora aparece nas conexões e nos padrões de comunicação entre esses microsserviços.

Como afirma a Lei de Conway[1], as organizações desenham sistemas que espelham suas estruturas de comunicação. Isso é bastante óbvio, já que se a arquitetura do seu sistema atravessa essas estruturas de comunicação, o esforço extra de comunicação entre diferentes equipes muitas vezes as leva a redesenhar toda a estrutura do sistema para que cada equipe tenha sua própria área de responsabilidade na arquitetura do sistema (por exemplo, um microsserviço), ou reestruturar as equipes para alinhá-las aos componentes da arquitetura a fim de reduzir essa sobrecarga de comunicação.

[1] https://en.wikipedia.org/wiki/Conway%27s_law

No entanto, do ponto de vista da agilidade, essas equipes de componentes nem sempre são ideais. Se você tiver, digamos, uma arquitetura de três camadas com um *front-end* Web, uma camada lógica de negócio e um banco de dados, poderá ser tentado a ter três equipes correspondentes. No entanto, muitas funcionalidades exigirão que as três equipes colaborem, pois essas funcionalidades geralmente precisam de algo na interface, alguma lógica de negócio e algum armazenamento. Isso fará com que as equipes tenham que esperar umas às outras e, portanto, reduz sua velocidade de mudança e agilidade.

Os métodos ágeis, portanto, tendem a favorecer equipes de funcionalidades interdisciplinares, que consistem em especialistas em cada uma das partes da arquitetura que são potencialmente tocadas por uma funcionalidade e incluem, também por exemplo, especialistas em UX, testes e DevOps. Essas equipes de funcionalidade podem lidar com uma funcionalidade de ponta a ponta. Isso é feito às custas de alguma eficiência. Além disso, se algum conhecimento especializado (por exemplo, em segurança, tecnologia de banco de dados etc.) for escasso e necessário para várias equipes, você terá limitações.

Uma vez que ambas as abordagens têm seus prós e contras, a maioria das organizações tende a ter uma combinação de equipes de funcionalidades e de componentes com base nas especificidades de suas soluções. Decidir o que funciona melhor depende fortemente da sua arquitetura.

Modelando Funcionalidades e Componentes no ArchiMate

Com modelos ArchiMate, você pode atender a ambas as abordagens. Na verdade, a subdivisão entre a estrutura ativa e os aspectos de comportamento no ArchiMate é bastante semelhante à existente entre componentes e funcionalidades. No final, você precisa de ambos, mas você pode optar por começar com um ou por outro na organização do trabalho. Abaixo, você vê o framework completo do ArchiMate, com um exemplo de camada de aplicativo que ilustra esses aspectos.

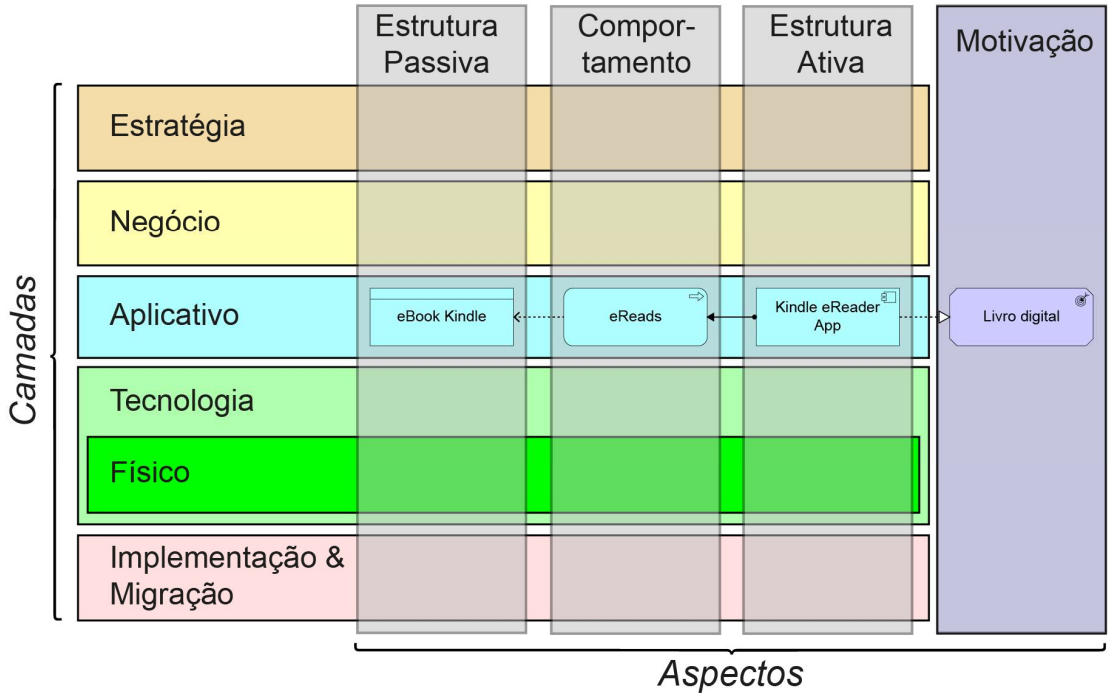

Figura 7. Framework do ArchiMate

Para uma solução nova a ser desenvolvida, você normalmente começa a pensar a partir de uma perspectiva de funcionalidade. No ArchiMate, você começa com conceitos de motivação para capturar as metas, requisitos e resultados planejados para a empresa e seus acionistas.

Em seguida, vem os conceitos de comportamento do ArchiMate, com serviços, funções e processos. Como explicado no padrão, um serviço de aplicativo é explicitamente definido como comportamento exposto do aplicativo que é significativo do ponto de vista do ambiente. Em resumo, ele representa o que um aplicativo faz para os seus usuários. Esse é o conceito-chave para a modelagem de funcionalidades de software no sentido ágil. Você não deveria confundir isso com o uso de 'serviço' em sentido técnico, por exemplo, como em um microsserviço, que na realidade é um tipo de componente de software. Em vez disso, a noção de serviço do ArchiMate tem um embasamento orientado para o negócio, na noção de uma parte que fornece um serviço para outra.

As funções e os processos do aplicativo são usados a seguir para modelar o comportamento interno necessário para fornecer esses serviços, ou seja, o que acontece nos bastidores. Quais componentes de aplicativo (ou papéis e atores de negócio, no caso de uma parte da solução não baseada em TI) executam esse comportamento e, por meio de quais interfaces você pode obter esses serviços, são as etapas finais desse processo de desenho.

Ao criar modelos de uma solução existente, muitas vezes é mais fácil começar com os componentes e suas interfaces, uma vez que são muitas vezes mais 'visíveis' e concretos do que serviços e funções. Este último é um pouco mais abstrato e pode exigir uma análise mais profunda desses componentes. Dado que a maioria das soluções são evolutivas e se inserem num panorama mais amplo de sistemas existentes, na prática você normalmente utilizará uma combinação de modelagem comportamento-para-estrutura e estrutura-para-comportamento.

Suportando a Colaboração em Equipes Ágeis com Modelos de Arquitetura

Seja qual for a forma como as equipes ágeis são estruturadas, você necessariamente separa algumas preocupações e combina algumas outras. Conhecer e entender essas dependências é fundamental em qualquer configuração de múltiplas equipes. Você quer evitar dividir coisas que estão estreitamente associadas, ou seja, possuem um alto grau de dependência, entre diferentes equipes. Isso resultaria em uma sobrecarga de comunicação desordenada entre essas equipes. E conhecer essas dependências é, obviamente, também o primeiro passo para reduzi-las.

No entanto, independente da forma como você estrutura suas equipes e sua arquitetura, em todos os casos há uma necessidade de comunicação que aumenta com o tamanho da organização (e do sistema). Por um lado, esta comunicação é 'vertical', para transmitir a visão corporativa de modo que as equipes se alinhem na mesma direção, e para dar retorno e contribuir com ideias inovadoras para cima na cadeia de comando, desde o desenvolvimento até a alta direção. Uma linha de visão clara ajuda a garantir que as metas sejam claras para todos os envolvidos e que as prioridades estejam alinhadas com essas metas. Embora as organizações possam usar todos os tipos de ferramenta para acompanhar suas iniciativas de desenvolvimento, a modelagem de arquitetura é a maneira mais eficiente para ilustrar as conexões entre estratégia, capacidades e itens do portfólio de desenvolvimento.

Vice-versa, o retorno sobre a viabilidade e a conveniência dessas metas vem da 'mão na massa' nas equipes ágeis, que estão em contato direto com os usuários, conhecem as possibilidades e limitações da tecnologia e fornecem melhorias, inovações e refinamentos para se adaptarem às circunstâncias locais. Os modelos de arquitetura fornecem tal conexão inequívoca entre a motivação e a realização, e permitem que você rastreie as dependências e os efeitos das mudanças nas prioridades, no planejamento etc.

Igualmente importante é a comunicação 'lateral' entre as equipes que trabalham em partes da solução e que precisam se comunicar, fornecer uma experiência de usuário integral, compartilhar recursos e, em geral, dependem umas das outras. Uma técnica comum para desenhar essas dependências é usar cordões de lã vermelha em um quadro Kanban ou Scrum, mas isso não escala para equipes maiores e para 'equipes de equipes' (espalhar esses cordões por todo o seu andar de trabalho de um quadro para outro seria um exercício interessante...). Além disso, naturalmente você não pode usar isso facilmente em nenhuma configuração remota. Novamente, os modelos de arquitetura são úteis aqui, mostrando claramente como diferentes componentes estão interconectados, como os processos de negócio dependem dos sistemas de TI, quais usuários estão trabalhando com quais aplicativos, onde os dados estão armazenados etc.

Figura 8. Dependências em um quadro Scrum (fonte: blog.xebia.fr)

Para ambas as equipes de componentes e de funcionalidades, saber quem está trabalhando em quais partes de uma solução é essencial para evitar conflitos, lacunas, sobreposições e retrabalho. Normalmente, cada equipe é totalmente responsável pelo funcionamento interno do que eles criam, ou seja, sua arquitetura e desenho de software. Eles podem usar o que quiserem para desenhar isso, desde esboços em quadros brancos até alguns modelos simples do ArchiMate, e equipes diferentes podem trabalhar de forma diferente. Como essas partes se relacionam umas com as outras é onde as coisas deveriam ser padronizadas de alguma forma, para que todas as equipes tenham o mesmo entendimento.

Uma visão geral de sua arquitetura em termos dos principais elementos de estrutura e comportamento (ou seja, componentes e funcionalidades) e de suas dependências ajuda a definir a estrutura das equipes e a apoiar a comunicação entre elas. É aqui, por exemplo, que um conjunto de visões ArchiMate da comunicação entre os componentes de aplicativo em seu panorama pode ser um artefato central de desenho. Além disso, gerenciar e compartilhar isto por meio de uma ferramenta apropriada é muito mais conveniente do que espalhar cordões por todo o seu espaço de trabalho. Um exemplo de um cenário de arquitetura como esse é mostrado abaixo, anonimizado e um pouco simplificado, a partir de um exemplo real de cliente no domínio de seguros.

A figura destaca o impacto de uma nova funcionalidade relacionada com a submissão on-line de sinistros de seguro. Isso afeta o website, o sistema INSUR e seus subsistemas, e vários fluxos de informação. A imagem mostra por meio de cores quais equipes Agile/DevOps são responsáveis por quais partes deste panorama, para que você saiba quem terá que colaborar nesta funcionalidade específica.

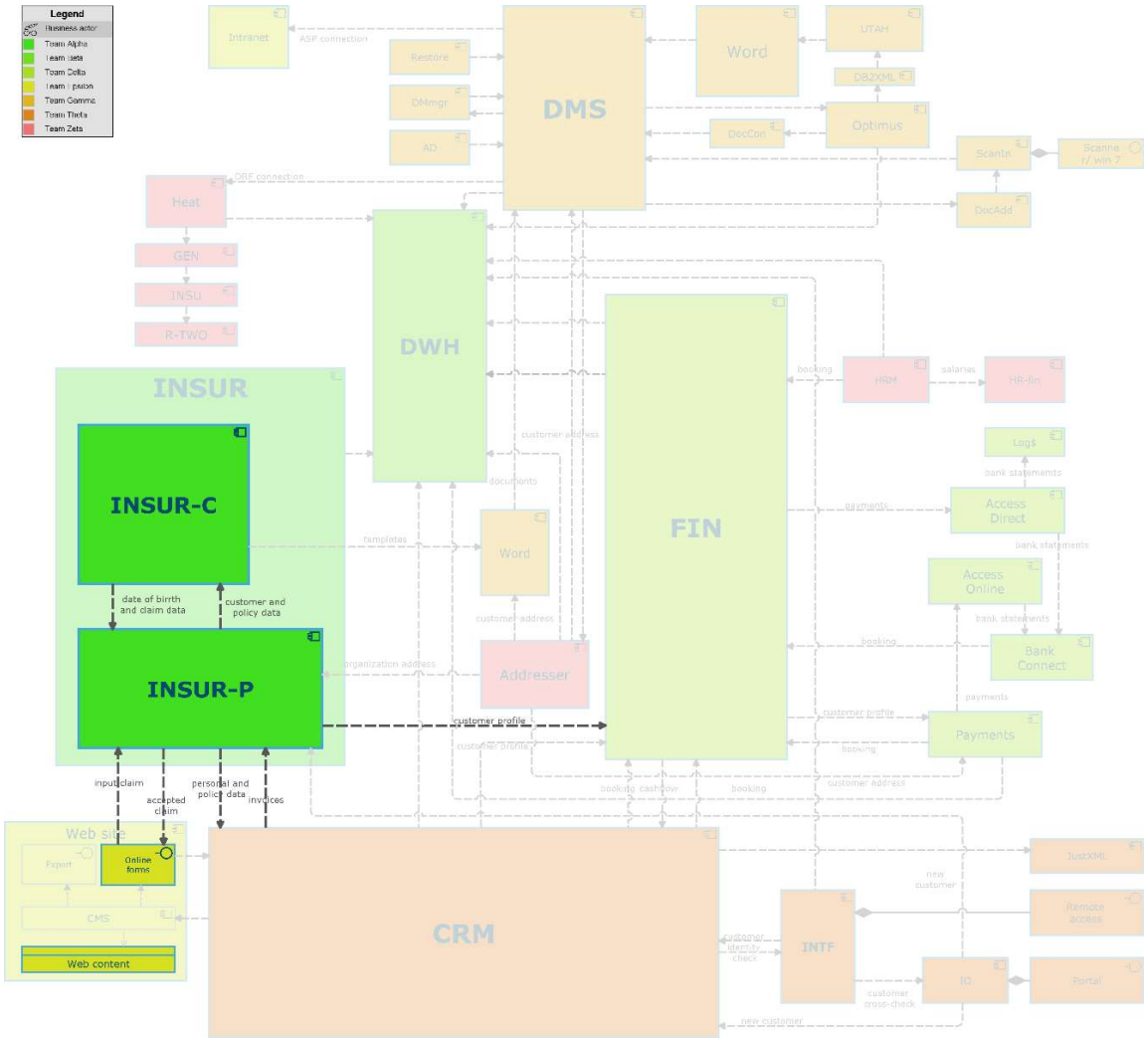

Figura 9. Panorama de aplicativos, destacando o impacto da funcionalidade, colorindo a responsabilidade da equipe de DevOps

Agora, o exemplo acima assume uma visão por componentes do panorama, e tem equipes para os principais sistemas e seus satélites, portanto, essa visão talvez seja mais relevante se você tiver equipes por componentes. No caso de equipes por funcionalidades, focar primeiro nos serviços de aplicativo que fornecem essas funcionalidades para os usuários pode fazer mais sentido. Mergulhar nesses serviços de aplicativo leva você às funções de aplicativo necessárias para fornecê-los, e em seguida você pode identificar quais componentes de aplicativo estão envolvidos na execução dessas funções.

Usar um modelo de arquitetura também pode ajudar equipes ágeis de várias outras formas, como:

- Identificar os efeitos das decisões de planejamento no nível da equipe sobre o resultado geral, por exemplo, adiar alguma história de usuário para um *sprint* posterior pode afetar a data de lançamento da funcionalidade de uma outra equipe.
- Apoiar a tomada de decisões entre as equipes a partir de uma perspectiva ponta a ponta, por exemplo, para garantir uma experiência unificada do usuário, evitar problemas de desempenho, solucionar problemas de segurança e conformidade etc.
- Compartilhar recursos escassos, tais como conhecimentos técnicos especializados.

O Nível Correto de Detalhe

Provavelmente, a pergunta mais frequente na modelagem é: "Qual é o nível correto de detalhe para os meus modelos?" E, num contexto ágil, talvez isso seja perguntado com ainda maior frequência. No entanto, isso não é algo que você pode simplesmente definir antecipadamente. Não há nenhum nível de detalhe único e ‘correto’. Isso depende fortemente do contexto e dos objetivos do seu esforço de modelagem.

O Contexto é Tudo

É claro que não é uma boa ideia documentar tudo em detalhes excruciantes. Os modelos precisam ser precisos o suficiente para a finalidade pretendida. Por exemplo, para ajudar os membros da equipe a entender uma solução em andamento; os especialistas em DevOps a implantá-la e resolver problemas (pense também em testes de integração, por exemplo); outras equipes a reutilizar componentes e serviços; e talvez o mais importante, qualquer um que precise fazer mudanças no futuro sem ter acesso aos desenhistas e desenvolvedores originais. É aqui que a intenção da arquitetura é especialmente importante.

Não modele mais do que você pode manter

Ter modelos que são muito detalhados muitas vezes leva a problemas na manutenção deles, simplesmente porque você não pode sustentar o esforço necessário para manter tudo atualizado. Isso, por sua vez, leva a modelos que envelhecem rapidamente e, às vezes, as pessoas se afastam completamente da modelagem simplesmente porque os modelos com os quais precisam trabalhar estão sempre desatualizados. É claro que este não é o caminho correto.

Portanto, você nunca deve modelar mais do que é capaz de manter (a menos que seja um modelo isolado e descartável). Mas como você decide, então, sobre o nível de detalhe correto?

Aproxime-se do nível 'correto' de forma iterativa

Há duas abordagens complementares a serem adotadas. Em primeiro lugar, chegar a um nível bom e útil de detalhe é um processo iterativo em si mesmo. Você pode usar o próprio processo ágil para melhorar isso. Em uma retrospectiva, você pode discutir isso e, portanto, refinar sua abordagem. Você tem as informações certas no momento certo? O esforço necessário para criar ou atualizar seus modelos era viável e valeu a pena? Como qualquer outra atividade em uma abordagem ágil, a modelagem deve estar sujeita a essa melhoria contínua na sua maneira de trabalhar.

Use uma abordagem baseada em risco

Em segundo lugar, você pode usar uma abordagem baseada em risco para decidir onde precisa de mais detalhes. A maioria dos métodos ágeis não é muito específica em relação a isso, mas alguns de seus predecessores, como o Processo Unificado (RUP-Rational Unified Process) são explicitamente focados em riscos e visam endereçar os riscos mais críticos logo no início do processo. De acordo com isso, você deveria criar modelos mais detalhados para as áreas de uma arquitetura ou projeto onde os riscos são maiores e menos detalhados onde você pode decidir rapidamente e as consequências de um erro são limitadas. Por exemplo, decidir sobre uma tecnologia de infraestrutura é frequentemente de alto risco, devido ao custo envolvido e para evitar o tempo e o esforço que serão necessários para reverter essa decisão caso se torne uma escolha errada. Outras áreas comuns de alto risco são proteção & saúde, segurança e conformidade. Pense nos riscos de informações de identificação pessoal (IIP), tratamento de transações financeiras, controle de processos de manufatura etc.

Nenhum nível único de detalhe

Isso também implica que seus modelos não terão um nível de detalhe único e semelhante em todas as situações. Mais ainda, a simples tentativa de visar um nível de detalhe tão padronizado pode levar você a dispender esforços onde não é realmente necessário, e inversamente, não dispor de tempo para as áreas onde isso realmente conta. Por isso, qualquer abordagem de modelagem que tente dizer aos modeladores antecipadamente qual é o nível de detalhe 'correto' irá, quase sempre, resultar em modelos com um nível de detalhe 'errado' – demasiadamente granular ou demasiadamente genérico.

Criando Modelos de Arquitetura de Maneira Ágil

As seções anteriores discutiram o uso de modelos de arquitetura no contexto do desenvolvimento ágil e o nível 'correto' de detalhe para esses modelos. Mas como você cria modelos em um contexto ágil?

Em primeiro lugar, o próprio processo de modelagem é algo que deve ser feito usando os mesmos princípios ágeis que você usa para desenvolver *software*. Isso significa que a criação de modelos é, por si só, um processo iterativo que começa pequeno e estende gradualmente os modelos com conteúdo necessário para fins específicos. Esse processo deveria se mover em estreito sincronismo com a evolução daquilo que esses modelos descrevem.

Definindo o futuro

Alguns modelos são usados como entrada para o desenho de algum *software* (ou processo de negócio, produto etc.) e, portanto, precisam estar 'um passo à frente'. Como sabemos pelas formas ágeis de trabalhar, você deseja adiar a tomada de decisões para o último momento possível, quando tiver as melhores informações disponíveis. Isso também é válido para modelos em um contexto ágil. Eles são desenvolvidos com uma mentalidade 'tempestiva', por exemplo, para fornecer contexto para os desenvolvedores nos próximos *sprints*, ou para ajudar os gerentes de portfólio a decidir sobre as prioridades para os próximos épicos etc. Isso significa também que quanto mais adiante um modelo pretende olhar, menos concreto ele será. Nos termos do ArchiMate, isso se traduz em:

- modelos que fornecem apenas a intenção de longo prazo e alto nível por meio dos elementos de Motivação e Estratégia;
- modelos de médio prazo e nível intermediário com detalhes mais concretos, por exemplo, para expressar as principais funcionalidades como serviços, e para definir conceitos como uma Pista Arquitetural (Architectural Runway);
- modelos detalhados de curto prazo que são usados, por exemplo, para descrever componentes de aplicativo e seus serviços e interfaces para desenvolvedores que precisam se conectar a eles.

Todos esses modelos deveriam ser criados por e com aqueles que estão diretamente envolvidos, de modo que a intenção e as ideias sejam capturadas na fonte.

Compreendendo o presente

Uma segunda categoria de modelos é aquela em que eles são criados para compreender a situação atual pré-existente. A maior parte do desenvolvimento é 'evolutivo' e você precisa entender esse contexto. Modelos são úteis para lidar com a complexidade desta situação atual. Por exemplo, padrões de comunicação complicados entre microsserviços são muito difíceis de entender apenas observando o código. Um modelo pode ajudar a esclarecer isso.

Agora, ter que criar esses modelos da situação atual a partir do zero pode ser um esforço desordenado. Idealmente, esses modelos teriam sido criados e mantidos por equipes anteriores, mas isso muitas vezes não é o caso. A segunda melhor opção é automatizar parte da criação desses modelos. Existem mais e mais ferramentas para descobrir, por exemplo, componentes de *software* que se comunicam na sua rede, e estas informações podem ser usadas para criar uma visão de baixo para cima do estado atual. Criação automatizada de, por exemplo, modelos ArchiMate da infraestrutura de TI com base em informações dessas ferramentas de descoberta, muitas vezes vinculadas a um *software* CMDB moderno, é como as equipes mais avançadas trabalham atualmente. No nível do negócio, as ferramentas de mineração de processos podem ser usadas para descobrir os processos de negócio de uma organização. E podemos esperar cada vez mais análises de dados e técnicas baseadas em inteligência artificial que ajudam a obter esses tipos de percepções e a construir seus modelos.

Entretanto, o que você não pode reconstruir apenas olhando o que existe no mundo lá fora é a intenção por trás do que você vê. Quais foram as principais escolhas e decisões de desenho? Quais alternativas foram rejeitadas e por quê? Quais foram as principais partes interessadas e quais foram os seus motivadores? Esse é o principal motivo pelo qual você precisa documentar o que você desenha e não apenas tentar fazer engenharia reversa após o fato. Este é novamente um argumento para modelar a arquitetura intencional em modelos (ArchiMate).

Modelando o que foi criado

Isso nos leva ao terceiro tipo de modelos, aqueles que são destinados a documentar o que foi criado após o fato, para que outros o entendam. Esses modelos são normalmente ‘um passo atrás' no processo de desenvolvimento e realização. Os métodos ágeis recomendam documentar tarde (mas não tarde demais!), de forma que você capture o que realmente foi realizado e possa incorporar experiências de aprendizagem relevantes, como o retorno dos clientes e as análises de colegas ou das retrospectivas. Mais ainda, a solução realizada pode se desviar da solução desenhada, e essas diferenças são importantes para entender e discutir todo o processo e os seus resultados.

O ciclo de vida dos modelos

Todos esses três tipos de modelo são úteis, mas devem ser criados e tratados de maneiras diferentes. Quanto mais tempo um modelo precisar viver, mais completo será o seu desenvolvimento e manutenção. Como argumentamos em uma postagem anterior, não modele mais do que você pode razoavelmente manter. Informações desatualizadas são muitas vezes mais perigosas do que nenhuma informação.

E há um quarto tipo de modelo não discutido aqui, de modelos temporários criados durante o processo de desenho criativo. Esses modelos geralmente têm uma vida útil curta e servem principalmente para inspirar, criar novas ideias, explicar e deixar que seus colegas validem suas ideias.

Entregar Valor Rapidamente

Abaixo você pode encontrar alguns ponteiros importantes para ter em mente para minimizar o tempo para o valor e entregar retorno frequente e iterativo sobre o investimento através da arquitetura corporativa. Nós ilustramos com modelos ArchiMate para lhe oferecer uma demonstração real daquilo que descrevemos na seção anterior.

Priorizar o Trabalho

Para priorizar com base no valor de negócio, você pode começar especificando temas estratégicos no seu modelo arquitetural. A visão da Contribuição das Metas na Figura 10 mostra dois temas estratégicos em uma seguradora, modelados como metas no ArchiMate (a parte de cima da figura). Vários resultados (na parte de baixo da figura) contribuem para estas duas metas estratégicas. Resultados representam efeitos produzidos e podem, por sua vez, ser ligados a épicos e habilitadores, que são elementos pivotais no SAFe e no desenvolvimento ágil em geral.

Figura 10: Ponto de Vista da Contribuição das Metas

Neste modelo, dois resultados têm forte influência positiva sobre a meta *Melhorar o serviço ao cliente*. Eles são um formulário de sinistro online e um aplicativo móvel para comunicação com corretores virtuais de seguro. Este último também contribui para a meta de *Reduzir o custo do trabalho*, um benefício adicional. Um novo algoritmo para a automatização do tratamento de sinistros contribui para a mesma meta.

A clareza do modelo permite aos arquitetos derivar instantaneamente uma priorização preliminar dos resultados a partir desta representação. Neste caso, eles deduzem que investir na capacidade de *Agente Virtual* produzirá o maior valor de negócio para a seguradora, e assim sendo este deveria ser priorizado à frente dos outros resultados.

Subdividindo Resultados

Para atingir uma priorização no nível corporativo, de cima para baixo, você deve levar em consideração fatores adicionais, tais como dependências arquiteturais dentro e através dos resultados, ou a atribuição de épicos através das equipes ágeis que são responsáveis pelos elementos arquiteturais distintos. A Arquitetura Corporativa se prova ser muito útil para isso, uma vez que a informação arquitetural fornece uma âncora sólida para este tipo de planejamento de alto nível.

Assim sendo, é proveitoso subdividir os resultados de uma forma consciente da arquitetura. Não há a necessidade de uma especificação detalhada de todos os requisitos. O objetivo é atingir uma granularidade apenas suficiente para delinear a impressão organizacional e arquitetural de cada resultado. Esta percepção sobre a impressão arquitetural do resultado é a chave para encontrar conflitos e gargalos. Tal informação normalmente não está disponível no seu gerenciamento típico de equipes ágeis ou em ferramentas de rastreamento como o Jira.

No exemplo da seguradora, o resultado *Interação do Cliente via Agentes Virtuais* pode ser dividido em dois requisitos que apresentam diferentes impressões arquiteturais. Tanto o componente Aplicativo Android como o componente Retaguarda Móvel são afetados, e eles são cobertos por equipes ágeis diferentes (veja a Figura 11 abaixo).

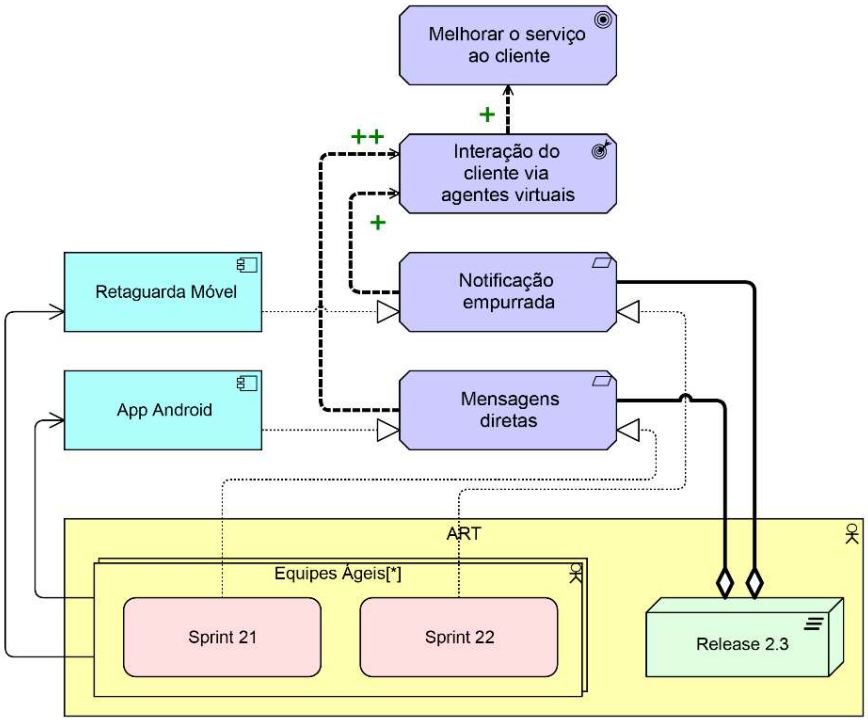

Figura 11. Alocação de recursos de desenvolvimento

De acordo com o planejamento de *sprints* da equipe ágil responsável, os requisitos são realizados durante *sprints* diferentes. Com base no roteiro do Trem de Entrega Ágil (*ART-Agile Release Train*), ambos os requisitos serão parte do mesmo *release*. O modelo ilustra claramente como o valor de negócio é criado ao longo do tempo e como os recursos de desenvolvimento são alocados.

Combinando a Arquitetura Corporativa e os Métodos Ágeis na Plataforma HoriZZon

Nós, na Centus e na BiZZdesign, temos pensado bastante em como podemos baixar as barreiras para a produtividade no local de trabalho, para permitir que os nossos clientes criem mais valor – isso inclui ajudá-los a integrar os Métodos Ágeis na sua prática de Arquitetura Corporativa com sucesso. Existem várias formas pelas quais nós suportamos um modelo mental e uma prática de arquitetura contínua. Para uma apresentação em profundidade da plataforma HoriZZon, por favor entre em contato conosco, pois mencionaremos brevemente apenas algumas delas aqui.

O HoriZZon foi construído em torno de um *conceito de fluxo imutável de eventos*. Isso significa que todas as mudanças no banco de dados são registradas como um fluxo de fatos que não mudam com o tempo. A vantagem de preservar este histórico é que isso permite que os usuários criem visões detalhadas para audiências específicas rápida e efetivamente. As visões são derivadas a partir dos dados através do processamento de todas as mudanças que ocorreram até um momento desejado no histórico de eventos registrado.

Em contraste, os *frameworks* de arquitetura historicamente focam em conjuntos de partes interessadas e nos pontos de vista que eles precisam em diferentes níveis de detalhes, cobrindo todos os domínios da arquitetura. Ao invés disso, nossa abordagem é compatível com o modelo mental dos métodos ágeis, uma vez que ela permite que os arquitetos busquem uma arquitetura “apenas suficiente, no momento certo”, e ofereçam para as partes interessadas as visões exatas que eles precisam, em tempo real.

Isso marca uma mudança de paradigma de um modelo solicitação-resposta para um modelo de subscrição-notificação, por meio do qual partes interessadas individuais podem receber fluxos personalizados de notícias de trabalho, comentar, fornecer retroalimentação e, em geral, colaborar em tempo real em torno da informação da arquitetura e do desenho. Mais ainda, vários outros tipos de fluxos de dados podem ser integrados e correlacionados com os seus modelos, desde custos e indicadores de desempenho até a satisfação do cliente e a conformidade regulatória.

Esta integração com fontes de dados externas inclui também ferramentas ágeis, como o Jira e as APIs do HoriZZon. Esta informação pode ser agregada em vários níveis e compartilhada por meio do portal HoriZZon. A plataforma permite que os arquitetos, desenvolvedores, e outros, criem várias visões de cores, roteiros e painéis de controle para informar às partes interessadas sobre o progresso das mudanças e o desempenho das equipes ágeis e dos trens de liberação. As decisões e o planejamento de transições podem ser baseados na informação disponível sobre o progresso, disponibilidade prevista de novas funcionalidades, dependências entre estas funcionalidades, e os resultados de negócio associados.

Os usuários podem calcular os efeitos dos atrasos antecipadamente e avaliar vários cenários quando os épicos ou funcionalidades precisarem ser repriorizados. Isso torna a mudança mais previsível e menos arriscada, uma vez que as partes interessadas focam nas necessidades verdadeiras do negócio.

Além disso, a plataforma HoriZZon oferece muito mais funcionalidades, incluindo o suporte para padrões e *frameworks* de melhores práticas, um ambiente de modelagem intuitivo, análises profundas, bem como mapas de jornada do cliente e capacidades de gerenciamento do ciclo de vida dos ativos. Nós não vamos discutir isso tudo, mas se você quer aprender mais sobre como integrar o Enterprise Studio e os Métodos Ágeis usando a plataforma HoriZZon, não hesite em nos contactar.

Conclusões

A premissa de que a arquitetura tem um lugar ao lado dos métodos ágeis é considerada com uma boa dose de ceticismo por boa parte dos profissionais da indústria. Isso impede que muitas organizações atinjam todo o seu potencial. Na medida em que a adoção dos Métodos Ágeis aumenta, se torna imperativo que o negócio descubra a grande complementariedade entre a Arquitetura Corporativa e os Métodos Ágeis, e de que contribuições estratégicas valiosas eles estão se privando ao longo do processo, por ignorar a arquitetura. Nós acreditamos que uma visão "ou-um-ou-outro" prejudica a competitividade.

Ao invés, nós estamos convencidos de que, quando adequadamente integradas, as duas práticas podem aumentar significativamente a resiliência do negócio. A Parte 1 deste guia argumentou em favor da sua compatibilidade. A Parte 2 ofereceu casos de uso e exemplos de melhores práticas em torno da Arquitetura Corporativa e dos Métodos Ágeis. A Parte 3 mostrou como os modelos de arquitetura podem suportar o desenvolvimento ágil, tanto ao nível da equipe como para coordenar diferentes equipes, escopos, níveis de detalhe e escalas de tempo. Esperamos que juntas elas tenham atingido o objetivo deste guia de ajudar os arquitetos corporativos a se tornarem mais produtivos e entregar a promessa da Arquitetura Corporativa mesmo em ambientes Ágeis.

Sobre a BiZZdesign

A BiZZdesign é uma fornecedora líder de software e serviços de transformação empresarial baseada na Holanda. Fundada em 2000, como uma cisão comercial de um instituto de P&D, hoje a empresa possui presença global e é reconhecida pelos analistas de mercado como um líder de mercado. O principal produto da BiZZdesign, o Enterprise Studio, é utilizado pelas maiores empresas mundiais e organizações governamentais através dos cinco continentes, onde ele desempenha um papel fundamental na habilitação exitosa da mudança dos negócios.

Sobre a Centus

A Centus é uma empresa de consultoria de negócios e transformação empresarial baseada em Belo Horizonte. Fundada em 2013, a empresa é focada na disseminação de conhecimentos sobre arquitetura corporativa, gerenciamento de decisões, transformação de negócios e a linguagem ArchiMate. Nossos principais produtos são plataformas de gerenciamento de decisões e de modelagem e repositório de arquitetura corporativa, contando com a parceria e o apoio de empresas líderes nos seus mercados, como a BiZZdesign.

Para mais informações, por favor visite bizzdesign.centus.com.br ou www.bizzdesign.com.