GRUPO I – CLASSE V – PLENÁRIO
TC 010.663/2013-4
Natureza: Levantamento de Auditoria
Interessado: Tribunal de Contas da União
Unidades: Tribunal Superior do Trabalho (TST); Banco Central do Brasil (Bacen); Instituto do Patrimônio Histórico e Artístico Nacional (Iphan); Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira (Inep); Supremo Tribunal Federal (STF)

SUMÁRIO: LEVANTAMENTO DE AUDITORIA. CONHECIMENTO ACERCA DA UTILIZAÇÃO DE MÉTODOS ÁGEIS NAS CONTRATAÇÕES PARA DESENVOLVIMENTO DE **SOFTWARE** PELA ADMINISTRAÇÃO PÚBLICA FEDERAL. CUMPRIMENTO DOS OBJETIVOS. ARQUIVAMENTO.

RELATÓRIO

Transcrevo a seguir o relatório de levantamento elaborado no âmbito da Secretaria de Fiscalização de Tecnologia da Informação - Sefti (peça 177), registrado no Sistema Fiscalis sob o número 290/2013, aprovado pelos dirigentes daquela unidade técnica:

"***1. Apresentação***

Atualmente, diversos órgãos da Administração Pública Federal iniciam investimentos para adotar contratações de serviços de desenvolvimento de ***software*** *utilizando métodos ágeis, ainda pouco difundidos nacionalmente, principalmente entre instituições públicas.*

2. Uma vez que a Secretaria de Fiscalização de Tecnologia da Informação (Sefti) julga ainda não deter a ***expertise*** *necessária para atuar em fiscalizações que envolvam esse tipo de metodologia, esta unidade técnica propôs a realização de levantamento com vistas a conhecer as bases teóricas do processo de desenvolvimento de* ***software*** *com métodos ágeis, bem como conhecer experiências práticas de contratação realizadas por instituições públicas federais, objetivando o aprendizado desta secretaria em relação ao tema.*

3. Com esse intuito, a Sefti, juntamente com a Secretaria de Soluções de TI (STI) deste Tribunal, unidade que já possui certo conhecimento em métodos ágeis em decorrência de experiência em desenvolvimento de ***softwares*** *realizado internamente utilizando essa metodologia, identificou instituições públicas que possuem contratos cujo objeto é de interesse desta fiscalização e visitou algumas delas. As visitas serviram para colher informações, por meio de relatos informais, acerca da adoção do paradigma de desenvolvimento em comento em seus contratos, bem como inteirar-se do conteúdo dos instrumentos convocatórios que os originaram.*

4. Como resultado do levantamento realizado, este relatório apresenta conceitos que abrangem a essência das metodologias ágeis de desenvolvimento de ***software****, descreve as principais metodologias atualmente utilizadas pelas instituições públicas brasileiras visitadas durante a execução do levantamento, relata aspectos das contratações analisadas e, por fim, relaciona alguns riscos inerentes passíveis de materialização nas contratações com metodologias ágeis.*

2. Introdução

*5. Ao longo dos últimos anos, as contratações realizadas por instituições públicas federais para construção de sistemas informatizados têm se baseado em metodologias de desenvolvimento de **software** alicerçadas, em sua maioria, no Processo Unificado de desenvolvimento de sistemas e suas variações.*

*6. Observa-se, contudo, o aumento da popularidade do uso de metodologias ágeis de desenvolvimento de **software** no mercado nacional e internacional. Essa realidade, somada às insatisfações frequentes das contratações de serviços geradas pelo uso do modelo corrente, tem levado algumas instituições públicas a acreditarem que podem obter melhores resultados com o uso das metodologias ágeis. Nesse cenário, instituições públicas iniciaram investimentos nessa área, capacitando seus servidores e instituindo internamente em suas equipes de tecnologia da informação, quando possível, métodos ágeis para o desenvolvimento de suas soluções. Passado esse movimento inicial, algumas dessas instituições começaram a realizar contratações para desenvolvimento de **software**, seja para projetos específicos, seja para fábricas de **software**, fundamentadas em metodologias ágeis.*

*7. Com esse contexto e dada a falta de conhecimento aprofundado acerca do tema por esta Secretaria de Fiscalização de Tecnologia da Informação (Sefti) que possibilite bem desempenhar seu poder fiscalizador, urgiu buscar a **expertise** necessária para suas futuras atuações quando demandada.*

*8. Nessa esteira, previamente à realização desta fiscalização, membros da Sefti foram capacitados por meio de participação em curso presencial relativo ao **Scrum**, uma das metodologias ágeis mais adotadas atualmente em projetos de desenvolvimento de **software**. Em seguida, esta unidade técnica submeteu proposta ao Ministro José Múcio Monteiro visando à realização de levantamento para a aquisição de maior conhecimento acerca do tema, a qual foi prontamente deferida.*

*9. A submissão da proposta ao Relator decorreu do fato de o Banco Central do Brasil (Bacen), instituição integrante de sua clientela, ser uma instituição pública reconhecidamente possuidora de experiência no desenvolvimento de **softwares** por meio de métodos ágeis.*

10. Para complementar e enriquecer as informações deste levantamento, outras organizações públicas com experiência na execução de projetos utilizando métodos ágeis foram visitadas durante esta fiscalização.

*11. Unindo esforços conjuntos da Sefti e da Secretaria de Soluções de TI (STI) deste Tribunal, o presente levantamento foi executado, buscando-se absorver conceitos teóricos em diversas fontes, como publicações físicas e eletrônicas, e observar relatos baseados na vivência prática de algumas instituições públicas federais que investiram, nos anos recentes, na contratação de desenvolvimento de **software** com metodologias ágeis.*

*12. Como resultado do trabalho empreendido, na primeira parte deste relatório a equipe do presente levantamento consolidou diversas informações, definições e conceitos colhidos relativos a metodologias de desenvolvimento de **software**, ágeis ou não, de forma a construir o embasamento teórico necessário ao entendimento dos relatos das experiências vividas pelas instituições públicas visitadas. Na segunda parte, o relatório descreve aspectos técnicos das contratações realizadas pelas instituições públicas que receberam a equipe. Por último, este relatório ainda confronta os valores que norteiam as metodologias ágeis com os princípios que orientam a Administração Pública, bem como apresenta alguns riscos inerentes ao tipo de contratação em estudo.*

13. Impende ressaltar que este trabalho não tem o intuito de esgotar as discussões sobre as características e os riscos da utilização de metodologias ágeis mediante o arcabouço normativo brasileiro. Trata-se de uma visão geral e primeira sobre o assunto, que subsidia esta Secretaria a tratar com o tema de maneira mais precisa.

2.1. Deliberação que originou o trabalho

14. A presente fiscalização foi originada de proposta elaborada pela Secretaria de Fiscalização de Tecnologia da Informação (Sefti) no âmbito do TC 009.890/2013-0, com fins à

*realização de levantamento acerca da utilização de métodos ágeis nas contratações para desenvolvimento de **software** pela Administração Pública Federal.*

15. Em 19/4/2013, por meio de despacho, o Ministro-Relator José Múcio Monteiro autorizou a proposta formulada pela Sefti (peça 17).

2.2. Objetivo e escopo

*16. A presente fiscalização tem por objetivo descrever a definição da essência dos métodos ágeis, os quais se constituem em uma metodologia para o desenvolvimento de **softwares**. Adicionalmente, este trabalho objetiva descrever como algumas instituições públicas federais estão realizando contratações utilizando métodos ágeis, assim como apresentar alguns riscos associados a essas contratações.*

17. A análise dos contratos identificados no âmbito desta fiscalização, à luz da conformidade à legislação vigente, não faz parte do escopo deste trabalho.

2.3. Metodologia e limitações

*18. Inicialmente, em concordância com os padrões de levantamento definidos pelo TCU, por meio de pesquisas na internet e em virtude do conhecimento técnico da Secretaria de Soluções de TI (STI), a equipe de fiscalização identificou quatorze instituições públicas federais que potencialmente teriam contratos de desenvolvimento de **software** utilizando métodos ágeis. Dessas instituições, conforme o tempo existente para a fase de execução da fiscalização e utilizando o critério de localização geográfica, beneficiando as instituições com sede em Brasília/DF, foram selecionadas sete para visitação e entrevistas com os gestores dos contratos, a saber:*

18.1. Tribunal Superior do Trabalho (TST);

18.2. Banco Central do Brasil (Bacen);

18.3. Instituto do Patrimônio Histórico e Artístico Nacional (Iphan);

18.4. Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira (Inep);

18.5. Supremo Tribunal Federal (STF);

18.6. Departamento de Informática do Sistema Único de Saúde (Datasus); e

18.7. Empresa Brasileira de Serviços Hospitalares (EBSERH).

19. Das instituições visitadas, verificou-se que somente as cinco primeiras possuíam objeto de interesse para este trabalho. Diante disso, a equipe de fiscalização obteve, para análise, os respectivos editais de licitação.

20. Além dos órgãos supracitados, a equipe de fiscalização também se reuniu com o Serviço Federal de Processamento de Dados (Serpro), o qual foi contratado para desenvolver módulos do sistema Novo Siafi para a Secretaria do Tesouro Nacional utilizando métodos ágeis. Uma vez que, contratualmente, não foi definida a utilização da metodologia ágil, o contrato balizador do ajuste não foi analisado.

21. Como limitação para o presente trabalho, pode-se citar que, ainda que a equipe de fiscalização tenha obtido, em certa medida, conhecimento teórico e prático, o pleno conhecimento sobre as metodologias ágeis abrange um vasto domínio que não pôde ser atingido pela equipe durante a execução deste trabalho, devido ao reduzido período de tempo para a sua realização.

3. Visão geral do objeto

*22. O presente trabalho trata de levantamento acerca da viabilidade da adoção de metodologias ágeis de desenvolvimento de **software** por instituições públicas federais em suas contratações, considerando suas características e seus principais riscos mediante o arcabouço jurídico vigente. Para melhor entender o objeto desta fiscalização, inicialmente faz-se necessária a abordagem de conceitos teóricos e das origens de algumas metodologias de desenvolvimento de sistemas de informação que têm sido largamente utilizadas na história recente da engenharia de **software**.*

3.1. Metodologias de desenvolvimento de software

*Produto de **Software***

*23. A ISO/IEC 12207-1998 é uma norma técnica elaborada pela **International Organization for Standardization (ISO)** que trata de processos de ciclo de vida de **software**. Ela define conceitos e propõe uma estrutura comum para padronizar a discussão em torno de procedimentos, métodos, ferramentas e ambientes de desenvolvimento e de gerência de **software**.*

*24. Segundo a norma citada, produto de **software** é um conjunto de programas de computador, procedimentos e possível documentação e dados associados, enquanto serviço de **software** é a execução de atividades, trabalho ou obrigações relacionados ao produto de **software**, tais como seu desenvolvimento, manutenção e operação.*

*25. Para efeitos didáticos, neste texto tratar-se-á **software** como sinônimo de 'produto de **software**'.*

*Metodologia de desenvolvimento de **software***

*26. De acordo com o dicionário Aurélio, no campo da literatura, metodologia é definida como o conjunto de técnicas e processos utilizados para ultrapassar a subjetividade do autor e atingir a obra literária. Já o dicionário Webster, no ramo da computação, conceitua metodologia como um conjunto organizado de procedimentos e guias documentados para a execução de uma ou mais fases do ciclo de vida do **software**.*

*27. Alinhando-se a essas definições, em engenharia de **software**, uma metodologia de desenvolvimento comumente é entendida como um conjunto estruturado de práticas que pode ser repetível durante o processo de produção do sistema ou, ainda, a forma de se utilizar um conjunto de práticas, métodos ou processos para se desenvolver ou manter um produto de **software**, de modo que se evite subjetividade na execução do trabalho. O uso de metodologias visa à produtividade das equipes e à qualidade do produto.*

*28. As metodologias de desenvolvimento de **software** surgiram em um contexto tecnológico muito diferente do atual, baseado apenas em soluções para computadores de grande porte (**mainframes**) e terminais burros, época na qual o custo para se fazer alterações e correções nos sistemas era muito elevado. Isso se deu em decorrência do limitado acesso aos computadores e da inexistência de modernas ferramentas de apoio ao desenvolvimento.*

*29. Por sua vez, a definição de processo de **software** se assemelha à de metodologia de desenvolvimento de **software**, conforme se depreende da leitura combinada dos conceitos de processo e de **software** constante das normas NBR ISO/IEC 15504-1 e 12207: conjunto de atividades que se inter-relacionam ou que interagem entre si, que transforma entradas em um conjunto de programas de computador, procedimentos, possível documentação e dados associados.*

*30. Atualmente, modelos de melhoria de processos de **software**, como o CMMI e o MPS-Br, bem como a jurisprudência deste Tribunal (Acórdãos 953/2009, 1.233/2012, 3.132/2012 e 1.167/2013, todos do Plenário do TCU), têm utilizado o termo 'processo de **software**' em detrimento a 'metodologia de desenvolvimento de **software**', embora as definições de ambos sejam, em essência, idênticas.*

*31. De todo modo, este trabalho utilizará, excepcionalmente, o termo 'metodologia de desenvolvimento de **software**', no intuito de se aproximar da linguagem utilizada pelos especialistas em métodos ágeis, objeto deste trabalho.*

Metodologia em cascata ou clássica

*32. Nos anos 1970 difundiu-se internacionalmente uma das mais conhecidas metodologias de desenvolvimento de **software**, o modelo em cascata (**waterfall**), também conhecido como clássico ou linear. Nessa metodologia, largamente utilizada até o início da década de 1990, o sistema é todo planejado e documentado, seguindo diversas fases, sequencialmente, antes de ser implementado. As atividades do processo de desenvolvimento são estruturadas em uma ordem fixa de fases (cascata), na qual as saídas de cada fase – geralmente documentos – após aprovadas, tornam-se entradas para a próxima. Espera-se, no fim da última fase, que o produto de **software** esteja pronto, baseando-se na premissa de que o trabalho nas fases anteriores foi realizado de maneira correta e gerou os produtos que delas se esperava.*

*33. Embora tenha auxiliado a engenharia de **software** quando de seu surgimento, oferecendo uma forma disciplinada para a construção de soluções, com o advento de novas tecnologias, como microcomputadores utilizados em larga escala, o modelo em cascata passou a não atender às necessidades emergentes. Fomentaram-se, então, diversas críticas a esse modelo, como a impossibilidade de retornar de modo simples à execução de atividades em fases anteriores e a excessiva produção de documentação nas fases iniciais do projeto. Isso causava, respectivamente, uma elevação significativa do custo das mudanças no projeto e demora na disponibilização do **software**.*

Metodologia baseada em prototipação

*34. Frequentemente, dado o dinamismo dos problemas empresariais sobre os quais o sistema atuará, o cliente possui um conjunto de objetivos genéricos para o **software**, geralmente conhecidos, e um conjunto de detalhes específicos, não identificados minuciosamente no momento de sua concepção. Por outro lado, em alguns casos, o desenvolvedor necessita assegurar a eficiência de um algoritmo, a adaptabilidade de um sistema operacional ou a forma que a interação homem-máquina deve ter no sistema a ser construído.*

*35. Para auxiliar nessas questões, surgiu a prototipação, que consiste em um processo no qual o desenvolvedor pode criar um modelo do **software** que deve ser construído. O modelo pode assumir várias formas, inclusive a construção de um protótipo que implementa um subconjunto das funcionalidades requeridas do **software** desejado.*

*36. O processo de prototipação é iterativo, repetindo-se fases bem definidas, basicamente para colher e refinar requisitos. Portanto, o objetivo é construir o protótipo e obter o **feedback** do cliente em fases iniciais do projeto. O protótipo serve como mecanismo para identificar os requisitos do **software**, raramente servindo como produto final, uma vez que foi construído 'às pressas', sem considerar importantes aspectos do **software**, como sua qualidade geral a longo prazo, seu desempenho e sua usabilidade.*

*37. Por isso, a prototipação tem como grande problema a criação da expectativa do cliente que enxerga o protótipo como uma versão funcional do **software**, enquanto o mesmo foi fragilmente construído de modo a validar um conjunto de requisitos iniciais. Esse fato pode levar o desenvolvedor a comprometer a implementação visando disponibilizar um protótipo funcional rapidamente, sem se preocupar em desenvolver o produto em si nas fases posteriores.*

Metodologia Espiral

38. O modelo em espiral, proposto em 1988, foi desenvolvido para abranger as melhores características do modelo clássico (cascata) e do modelo de prototipação, adicionando, ao mesmo tempo, a análise de riscos, não presente nessas outras metodologias.

*39. Como o próprio nome referencia, o modelo é representado por uma espiral na qual, para o desenvolvimento do **software**, atividades de quatro fases (planejamento, análise de riscos, construção e avaliação do cliente) são realizadas linearmente, como no modelo em cascata, porém de forma iterativa, dando uma abordagem evolucionária ou incremental à engenharia de **software**. A cada iteração dessas fases, uma versão é construída e, à medida que as iterações avançam, as versões tornam-se mais completas, de forma progressiva.*

40. O modelo em espiral aproxima o cliente do processo de desenvolvimento ao permitir que, de sua avaliação do produto, surjam sugestões e modificações, isto é, a avaliação do cliente fundamenta as próximas etapas de planejamento e análise de riscos.

41. Essa metodologia apresenta como principal problema o risco da perpetuação do desenvolvimento, dado o seu caráter evolutivo.

Metodologia de Processo Unificado

*42. Em paralelo ao aparecimento do modelo clássico, no final da década de 1960, em projeto desenvolvido por Ivar Jacobson na empresa de telefonia **Ericsson**, surgiram as origens da metodologia de desenvolvimento denominada Processo Unificado (**Unified Process – UP**). Em 1987, ao sair da **Ericsson**, Jacobson fundou a empresa **Objectory Systems**, renomeada, em 1991, para*

Objectory AB. *Essa companhia despendeu anos de esforços para o desenvolvimento do* ***ObjectOry****, o qual consistia em um método de desenvolvimento de* ***softwares*** *orientado a objetos.*

43. Em 1995, com a experiência adquirida ao longo de seus projetos, Jacobson lançou o livro ***Object-Oriented Engineering****, no qual propunha a idéia de que os requisitos do cliente deveriam ser a força diretiva mais importante no desenvolvimento de* ***software****, além de mostrar a gênese da ênfase do Processo Unificado na arquitetura dos sistemas. Ainda naquele ano, a empresa* ***Rational Software Corporation*** *adquiriu a* ***Objectory Systems*** *e, no âmbito do produto* ***Rational Objectory Process (ROP)****, continuou o desenvolvimento do* ***ObjectOry****, expandindo-o para áreas ainda não abordadas, como ferramentas de gerenciamento e de desenvolvimento de projetos.*

44. À medida que o ***ROP*** *progredia, a empresa* ***Rational*** *continuava a adquirir empresas fabricantes de ferramentas de desenvolvimento de* ***software*** *ou a elas se fundir. As ferramentas adquiridas agregaram valor ao produto* ***ROP*** *e, em 1998, a* ***Rational*** *alterou o seu nome para* ***Rational Unified Process (RUP)****. Assim, passou a consistir em uma versão comercial do* ***UP****,* ***framework*** *ou metodologia de desenvolvimento que foi construída praticamente de forma simultânea ao* ***RUP****, também pela empresa* ***Rational****.*

45. Contrapondo-se ao modelo em cascata, o ***UP*** *propôs-se como uma metodologia de desenvolvimento iterativa e incremental, características herdadas do modelo espiral. No* ***UP****, as quatro fases de desenvolvimento do sistema (concepção, elaboração, construção e transição) são repetidas em ciclos, obtendo-se, ao término de cada ciclo, uma versão funcional do* ***software****. Em acréscimo, o* ***UP*** *também se propôs a ser adaptativo, de forma a atender às necessidades específicas de cada projeto.*

46. O surgimento do Processo Unificado trouxe diversas vantagens ao processo de desenvolvimento de ***softwares*** *quando comparado ao modelo em cascata. Algumas delas são: a entrega de versões com mais constância, precipitando o* ***feedback*** *sobre o produto; a antecipação em identificar as mudanças, diminuindo o custo das alterações no decorrer do desenvolvimento; o controle dos riscos inerentes ao projeto; o foco no produto; e o aumento da qualidade do produto final*

47. Contudo, as organizações que produzem e contratam projetos de construção de ***software****, na tentativa de adotar o* ***UP*** *e suas variantes, como o* ***RUP****, não raramente desvirtuam suas principais características (iteratividade e incrementalidade), e terminam por realizar adaptações nas quais a própria essência do modelo se perde. Nesses casos, as atividades de construção de* ***software****, que deveriam ser executadas várias vezes em vários ciclos, passam a ser executadas uma única vez, de maneira puramente sequencial, similar ao que ocorre no modelo em cascata, desvirtuando o conceito fundamental relacionado à geração incremental que agregue valor ao negócio.*

Metodologias Ágeis

48. Além das metodologias anteriormente citadas, atualmente está em voga a utilização de metodologias ágeis.

49. As metodologias ágeis são representadas por um conjunto de valores e princípios a ser utilizado no processo de desenvolvimento de sistemas. Esse conjunto de valores e princípios foi externado em 2001, por meio da divulgação do Manifesto para Desenvolvimento Ágil de ***Software****, criado por um grupo de dezessete especialistas em processos de desenvolvimento de* ***software*** *que, individualmente, já utilizavam práticas e teorias adaptadas.*

50. O Manifesto para Desenvolvimento Ágil de ***Software****, também conhecido apenas como Manifesto Ágil, foi alicerçado em um pequeno conjunto de princípios que, segundo os autores, pareciam ter sido recorrentemente respeitados nos casos de sucesso dos seus projetos.*

51. Embora alguns profissionais acreditem que a essência e as práticas adotadas pelas metodologias ágeis sejam modernas e inovadoras, pode-se dizer que sua origem remonta aos anos que sucederam à Segunda Guerra Mundial, no âmbito da empresa Toyota. Naquela época, a indústria japonesa tinha uma produtividade muito baixa e uma enorme falta de recursos, o que dificultava a adoção do modelo de produção em massa. Nesse contexto, a Toyota desenvolveu um sistema de

*produção chamado de **Lean Manufacturing** (produção enxuta) ou **Toyota Production System (TPS)** – Sistema de Produção Toyota, em português – cujo objetivo consistia em aumentar a eficiência da produção pela eliminação contínua de desperdícios. O **TPS** contemplava diversas características ou práticas incorporadas às metodologias ágeis, entre as quais se destacam:*

51.1. construir apenas o que é necessário, eliminando desperdícios;

51.2. realizar entregas rápidas e contínuas;

51.3. estar sempre aberto às mudanças.

*52. O Manifesto Ágil, transcrito a seguir, apresenta e descreve a essência de um conjunto de abordagens para desenvolvimento de **software** que vem sendo adotado desde a década de 1990.*

'Manifesto Ágil

*Estamos descobrindo maneiras melhores de desenvolver **software** fazendo-o nós mesmos e ajudando outros a fazê-lo. Através deste trabalho, passamos a valorizar:*

Indivíduos e interação entre eles mais que processos e ferramentas

***Software** em funcionamento mais que documentação abrangente*

Colaboração com o cliente mais que negociação de contratos

Responder a mudanças mais que seguir um plano

Ou seja, mesmo havendo valor nos itens à direita, valorizamos mais os itens à esquerda.'

53. Além dos valores expressos no manifesto, foram formulados os princípios que os sustentam, apresentados a seguir:

*54. Princípio 1: nossa maior prioridade é satisfazer o cliente, através da entrega adiantada e contínua de **software** de valor.*

*54.1. A priorização na construção das funcionalidades que o **software** deve possuir é feita com base na importância que elas têm para o cliente e no seu valor para o negócio, e não em outros critérios, como a complexidade da funcionalidade a ser implantada.*

*54.2. A entrega de versões intermediárias do **software** deve existir e ser incremental, antecipando ao máximo a sua utilização pelos usuários, a fim de aferir, pelo efetivo uso, se o seu funcionamento atende a suas expectativas. Essa característica também possibilita planejar melhor a construção dos requisitos seguintes, já que a equipe e os usuários possuirão uma experiência na utilização do que já está pronto.*

55. Princípio 2: aceitar mudanças de requisitos, mesmo ao fim do desenvolvimento. Processos ágeis se adéquam a mudanças para que o cliente possa tirar vantagens competitivas.

*55.1. A entrega frequente de partes do **software** em pleno funcionamento e o mais cedo possível possibilita que se construa um ambiente de constante **feedback** a respeito das necessidades do cliente. Observa-se que tal **feedback** normalmente fomenta mudanças recorrentes dos requisitos por parte do cliente.*

*55.2. Os processos ditos ágeis tratam a mudança nas necessidades do cliente como algo natural e necessário para a construção de um **software** adequado. Tanto melhor for a percepção do usuário na utilização real do sistema, formas melhores de resolver problemas ao longo do projeto podem ser idealizadas e construídas em conjunto.*

*56. Princípio 3: entregar **software** funcionando com frequência, na escala de semanas até meses, com preferência aos períodos mais curtos.*

56.1. Durante o projeto de desenvolvimento, recomenda-se que entregas parciais e funcionais sejam feitas com a maior frequência possível, para assegurar, o quanto antes, se o que está sendo construído realmente atende às necessidades dos clientes e dos usuários, mitigando, portanto, alguns riscos do projeto.

57. Princípio 4: pessoas relacionadas a negócios e desenvolvedores devem trabalhar em conjunto e diariamente, durante todo o curso do projeto.

57.1. No cerne desse princípio, está o acesso e a comunicação entre as pessoas da equipe que, independente do papel de cada uma, deve ser o mais simples possível. Ferramentas automatizadas e

encontros frequentes devem ser utilizados a fim de que a transferência de conhecimento não aconteça apenas por meio de produção e leitura de documentos, e sim por meio da comunicação informal.

57.2. Além disso, esse princípio materializa a posição de que o sucesso do projeto depende da dedicação e disponibilidade das pessoas envolvidas. Isso inclui uma maior participação do cliente em todas as fases do projeto, a fim de elucidar os requisitos, definir regras de negócio, dirimir dúvidas durante a construção das funcionalidades e avaliar prontamente o funcionamento dos ***softwares*** *intermediários que forem sendo construídos.*

58. Princípio 5: construir projetos ao redor de indivíduos motivados, dando a eles o ambiente e suporte necessário, e confiar que farão seu trabalho.

58.1. A motivação dos membros da equipe advém da disponibilização de um ambiente de trabalho saudável, constituído de ferramentas e instrumentos adequados para a execução do seu trabalho. Por sua vez, a confiança entre os membros da equipe é necessária para promover o ambiente colaborativo. E é no ambiente colaborativo que surgem soluções com qualidade.

59. Princípio 6: o método mais eficiente e eficaz de transmitir informações, para e por dentro de um time de desenvolvimento, é através de uma conversa 'cara a cara'.

59.1. A produção de documentos, planilhas, modelos e diagramas não deve substituir o diálogo contínuo entre o cliente e a equipe de desenvolvimento. É responsabilidade primordial da equipe técnica prover as melhores soluções tecnológicas e do cliente transmitir adequadamente as necessidades e regras do negócio. Segundo os princípios, por meio da colaboração entre os membros da equipe, aumentam consideravelmente as chances de se alcançar os objetivos do projeto.

60. Princípio 7: ***software*** *funcional é a medida primária de progresso.*

60.1. A principal medida de sucesso e progresso de um projeto de desenvolvimento de ***software*** *é ter o programa de computador em operação, com as funcionalidades sendo homologadas pelo cliente.*

60.2. Este princípio não dispensa a elaboração da documentação produzida pelas equipes, nem mesmo dá menos importância aos profissionais que não produzam programas de computador propriamente ditos. A proposta é que não se considere que o projeto está evoluindo com sucesso tendo como base somente a produção de documentação. Programas em funcionamento, em última instância, são o que definem o progresso do projeto de desenvolvimento de ***software****.*

61. Princípio 8: processos ágeis promovem um ambiente sustentável. Os patrocinadores, desenvolvedores e usuários devem ser capazes de manter, indefinidamente, passos constantes.

61.1. Recomenda-se que se mantenha, de forma equilibrada durante o projeto, o atendimento a todos os princípios. O sucesso dos projetos vem do equilíbrio e da constância na adoção de cada um dos princípios.

62. Princípio 9: contínua atenção à excelência técnica e ao bom ***design*** *aumenta a agilidade.*

62.1. Aumentar a agilidade significa, entre outras coisas, que o retorno do investimento em um projeto seja maximizado. Atentar-se à excelência técnica, em paralelo a todos os outros princípios, aumenta as chances de gerar um produto de qualidade e, por consequência, diminui os riscos na sustentação futura do ***software****. Assim, o retorno do investimento tende a se manter positivo durante todo o ciclo de vida do produto de* ***software****.*

63. Princípio 10: simplicidade, ou seja, a arte de maximizar a quantidade de trabalho que não precisou ser feito.

63.1. É importante que sejam desenvolvidas funcionalidades no ***software*** *exatamente como foram solicitadas, resistindo-se à tentação de desenvolver funcionalidades para atender a futuros problemas que ainda não existem.*

64. Princípio 11: as melhores arquiteturas, requisitos e ***designs*** *emergem de times auto-organizáveis.*

64.1. A auto-organização baseia-se na idéia de que é necessário horizontalizar as decisões de um time de alto desempenho. O modelo onde um gerente pensa de modo restrito nas tarefas de um projeto e as distribui para o restante da equipe não favorece a concepção de boas soluções. A ordem

se constrói com a participação ativa dos membros da equipe, independentemente do seu papel, e da construção de um senso de grupo. É baseado nessa mesma teoria que, hoje, ***softwares*** *de nível mundial são desenvolvidos de maneira livre (****open-source****), com colaboração remota de membros, sem que pessoas tenham que controlar quem faz o quê.*

64.2. Com o estudo deste princípio, observa-se que a gerência estrita perde lugar para a orientação, a definição e atribuição de tarefas passam a ser responsabilidade de todos, e as metas são do time e não mais individuais.

65. Princípio 12: em intervalos regulares, o time reflete sobre como se tornar mais efetivo. Então, ajustam-se e otimizam seu comportamento de acordo.

65.1. Nenhum processo de ***software*** *é perfeito. Recomenda-se que, de tempos em tempos, as pessoas reflitam sobre o processo, identificando pontos falhos e possíveis oportunidades de melhoria. A idéia é que o aperfeiçoamento periódico do processo permite a melhoria contínua no processo de trabalho, a fim de produzir* ***software*** *de qualidade.*

66. Nessa esteira, entende-se como Metodologia Ágil de desenvolvimento de ***software*** *o conjunto de métodos, processos e* ***frameworks*** *que são norteados pelos valores e princípios acima descritos.*

67. Com fins puramente didáticos, ao longo deste relatório, as metodologias não ágeis serão agrupadas em um conjunto que se convencionou denominar metodologias tradicionais.

3.2. Principais metodologias ágeis de desenvolvimento de software

68. Atualmente, várias metodologias ágeis são utilizadas pelo mercado mundial para o desenvolvimento de ***software****, tais como:* ***eXtreme Programming****,* ***Scrum****,* ***Feature Driven Development (FDD)****,* ***Dynamic Systems Development Method (DSDM)****,* ***Adaptive Software Development (ASD)****,* ***Crystal****,* ***Pragmatic Programming****,* ***Test Driven Development (TDD)****,* ***Kanban****, entre outras.*

69. No Brasil, segundo o Relatório Técnico RT MAC-2012-03 do Departamento de Ciência da Computação do Instituto de Matemática e Estatística da Universidade de São Paulo (IME-USP), de maio de 2012 (peça 18), fundamentado em pesquisa sobre a adoção de práticas em times e organizações brasileiras, as metodologias mais utilizadas são ***Scrum****,* ***eXtreme Programming (XP)*** *e a combinação entre elas. Uma vez que o trabalho em campo identificou que essas metodologias, acrescidas do* ***Kanban****, são as mais utilizadas nas instituições públicas visitadas, elas serão descritas sucintamente a seguir.*

3.2.1. Scrum

70. Em 1986, Takeuchi e Nonaka publicaram um artigo na revista ***Harvard Business Review****, descrevendo uma nova abordagem para desenvolvimento de produtos. Empresas como 3M e* ***Canon*** *inovaram na forma de desenvolver seus produtos, a fim de que o processo produtivo fosse mais rápido e os produtos chegassem ao mercado o quanto antes. A organização das equipes nessas empresas se assemelhava à formação* ***scrum*** *dos times nos jogos de* ***rugby****, com composição multidisciplinar, constante interação entre os membros, que trabalhavam juntos do início ao fim do processo produtivo para entregar um produto.*

71. Inspirados nesse artigo, Ken Schwaber e Jeff Sutherland desenvolveram o ***Scrum****, que consiste em um* ***framework*** *que pode ser utilizado para desenvolver e manter produtos complexos. Sua definição é encontrada no Guia do* ***Scrum*** *(peça 19), no qual seus criadores documentam como o* ***framework*** *foi desenvolvido e mantido por eles por mais de vinte anos. É livre e oferecido por seu Guia, que pode ser obtido gratuitamente no sítio eletrônico* ***www.scrum.org****. Por tal motivo, as definições e informações a seguir apresentadas, foram extraídas de seu próprio guia oficial.*

72. Segundo a visão geral do Guia, o ***Scrum*** *pode ser definido como:*

'Um ***framework*** *dentro do qual pessoas podem tratar e resolver problemas complexos e adaptativos, enquanto produtiva e criativamente entregam produtos com o mais alto valor possível.* ***Scrum*** *é:*

- Leve

- Simples de entender

- Extremamente difícil de dominar

***Scrum** é um **framework** estrutural que está sendo usado para gerenciar o desenvolvimento de produtos complexos desde o início de 1990. **Scrum** não é um processo ou uma técnica para construir produtos; em vez disso, é um **framework** dentro do qual você pode empregar vários processos ou técnicas. O **Scrum** deixa claro a eficácia relativa das práticas de gerenciamento e desenvolvimento de produtos, de modo que você possa melhorá-las.' (peça 19, p. 3)*

*73. O **Scrum** é fundamentado nas teorias empíricas de controle de processo, ou empirismo, segundo o qual o conhecimento vem da experiência e de tomada de decisões baseada no que é conhecido. Também emprega uma abordagem iterativa e incremental para aperfeiçoar a previsibilidade e o controle de riscos (peça 19, p. 4).*

*74. O **Scrum** consiste nas equipes ou times **Scrum**, as quais são associadas a papeis, eventos, artefatos e regras. Cada componente dentro do **framework** serve a um propósito específico e é essencial para o uso e sucesso da metodologia. Suas regras integram os eventos, papeis e artefatos, administrando as relações e interações entre eles (peça 19, p. 5).*

*Papeis do **Scrum***

*Time **Scrum***

*75. Um time **Scrum** é composto pelo **Product Owner (PO)**, pela equipe de desenvolvimento e pelo **Scrum Master**. Os times são auto-organizáveis e multifuncionais. Equipes auto-organizáveis escolhem a melhor forma para completarem seu trabalho, em vez de serem dirigidos por outros de fora da equipe. Equipes multifuncionais possuem todas as competências necessárias para completar o trabalho sem depender de outros que não fazem parte da equipe. O modelo de equipe é projetado para aperfeiçoar a flexibilidade, criatividade e produtividade (peça 19, p. 5).*

*76. Times **Scrum** entregam produtos de forma iterativa e incremental, maximizando as oportunidades de **feedback**. Entregas incrementais de produto 'pronto' garantem que uma versão potencialmente funcional do produto do trabalho esteja sempre disponível (peça 19, p. 5).*

Product Owner

*77. O **Product Owner (PO)**, ou dono do produto, é o responsável por maximizar o valor do produto e do trabalho da equipe de desenvolvimento. A maneira como isso é feito pode variar amplamente entre as organizações, times **Scrum** e indivíduos. O **PO** é a única pessoa responsável por gerenciar o **backlog** do produto, por meio de tarefas que incluem (peça 19, p. 5):*

*77.1. expressar claramente os itens do **backlog** do produto;*

77.2. ordenar ou priorizar esses itens para melhor alcançar as metas e missões;

77.3. garantir o valor do trabalho realizado pelo time de desenvolvimento;

*77.4. garantir que o **backlog** do produto seja visível, transparente, claro para todos, e mostrar no que o time **Scrum** trabalhará a seguir; e*

*77.5. garantir que a equipe de desenvolvimento entenda os itens do **backlog** do produto no nível necessário.*

*78. O **backlog** do produto é definido nos itens 106 a 110 deste relatório.*

*79. O **Product Owner** pode desempenhar essas atividades ou delegá-las para que a equipe de desenvolvimento as faça. No entanto, o **PO** continua sendo o responsável. Importa ressaltar que o **Product Owner** é uma pessoa e não um comitê. Ele pode representar o desejo de um comitê no **backlog** do produto, mas aqueles que desejarem uma alteração nas prioridades dos itens de **backlog** devem convencer o **PO** a tomar essa decisão. Para que o **Product Owner** tenha sucesso, toda a organização deve respeitar as suas decisões, que são visíveis no conteúdo e na priorização do **backlog** (peça 19, p. 5-6).*

Equipe de desenvolvimento

*80. A equipe de desenvolvimento consiste de profissionais que realizam o trabalho de entregar uma versão potencialmente utilizável que incrementa o produto 'pronto' ao final de cada **sprint**.*

*Somente integrantes da equipe de desenvolvimento criam incrementos do produto de **software** (peça 19, p. 6).*

81. As equipes de desenvolvimento são estruturadas e autorizadas pela instituição para organizar e gerenciar seu próprio trabalho. A sinergia resultante aperfeiçoa a sua eficiência e eficácia como um todo. Elas possuem as seguintes características (peça 19, p. 6):

*81.1. são auto-organizadas, isto é, ninguém (nem mesmo o **Scrum Master**) diz à equipe como transformar o **backlog** do produto em incrementos de funcionalidades potencialmente utilizáveis;*

81.2. são multifuncionais, possuindo todas as habilidades necessárias, enquanto equipe, para criar o incremento do produto;

*81.3. o **Scrum** não reconhece títulos para os integrantes da equipe que não seja o de desenvolvedor, independentemente do trabalho que está sendo realizado pela pessoa, não havendo exceções para esta regra;*

81.4. individualmente, os integrantes da equipe de desenvolvimento podem ter habilidades específicas e áreas diferentes de especialização, mas a responsabilidade pertence à equipe como um todo;

81.5. equipes de desenvolvimento não contém subequipes dedicadas a domínios específicos de conhecimento, tais como teste ou análise de negócios.

Scrum Master

*82. O **Scrum Master** é o responsável por garantir que o **Scrum** seja entendido e aplicado, de forma a assegurar que o time **Scrum** adere à teoria, práticas e regras da metodologia. É um servo-líder para o time. Ainda tem como atribuição ajudar aqueles que estão fora do time a entender quais as suas interações com o time são úteis e quais não são, objetivando maximizar o valor criado pelo time **Scrum** (peça 19, p. 7).*

*83. O **Scrum Master** trabalha para atender às necessidades do **Product Owner**, encontrando técnicas para o gerenciamento efetivo do **backlog** do produto; comunicando claramente a visão, objetivo e itens do **backlog** do produto para a equipe de desenvolvimento; ensinando o time **Scrum** a criar itens de **backlog** do produto de forma clara e concisa; compreendendo, a longo prazo, o planejamento do produto no ambiente empírico; compreendendo e praticando a agilidade; e facilitando os eventos **Scrum** conforme exigidos ou necessários (peça 19, p. 7).*

*84. Por sua vez, trabalha também para auxiliar a equipe de desenvolvimento, treinando-a em autogerenciamento e interdisciplinaridade; ensinando-a e liderando-a na criação de produtos de alto valor; removendo impedimentos para o seu progresso; facilitando os eventos **Scrum** conforme exigidos ou necessários; e treinando a equipe de desenvolvimento em ambientes organizacionais nos quais o **Scrum** não é totalmente adotado e compreendido (peça 19, p. 7).*

*85. Por fim, o **Scrum Master** também trabalha auxiliando a instituição, liderando e treinando a organização na adoção da metodologia; planejando implementações **Scrum** dentro da organização; ajudando funcionários e partes interessadas a compreender e tornar aplicável o **Scrum** e o desenvolvimento de produto empírico; causando mudanças que aumentam a produtividade do time; e trabalhando com outro **Scrum Master** para aumentar a eficácia da aplicação da metodologia na organização (peça 19, p. 7).*

*Eventos do **Scrum***

*86. Eventos prescritos são usados para criar uma rotina e minimizar a necessidade de reuniões não definidas. O **Scrum** usa eventos **time-boxed**, ou seja, 'espaços de tempo' nos quais todo evento tem uma duração máxima definida, garantindo que uma quantidade adequada de tempo seja gasta no planejamento, sem permitir perdas ao longo desse processo (peça 19, p. 8).*

*87. Além da **sprint**, que é um container para outros eventos, cada evento no **Scrum** é uma oportunidade formal para inspecionar e adaptar alguma coisa, sendo especificamente projetado para permitir transparência e inspeção criteriosa. A não inclusão de qualquer um dos eventos resultará na redução da transparência e na perda de oportunidade para inspecionar e adaptar (peça 19, p. 8).*

Sprint

*88. O coração do **Scrum** é a **sprint**, um **time-box** de um mês corrido ou menos, durante o qual uma versão incremental potencialmente utilizável do produto ('pronto') é criada. **Sprints** tem durações coerentes em todo o esforço de desenvolvimento. Uma nova **sprint** inicia-se imediatamente após a conclusão da anterior. São compostas por uma reunião de planejamento, reuniões diárias, trabalho de desenvolvimento, revisão e retrospectiva (peça 19, p. 8).*

*89. Quando o horizonte temporal da **sprint** é muito longo, a definição do que será construído pode mudar, a complexidade pode aumentar e o risco pode crescer. Com curtos períodos, permitem uma previsibilidade que garante a inspeção e adaptação do progresso em direção à meta pelo menos a cada mês corrido (peça 19, p. 8).*

*90. Durante a **sprint**, não são feitas mudanças que podem afetar seu objetivo, a composição da equipe de desenvolvimento permanece constante, as metas de qualidade não diminuem e o escopo pode ser esclarecido e renegociado entre o **Product Owner** e a equipe de desenvolvimento à medida que seja melhor entendido (peça 19, p. 8).*

*91. Cada **sprint** pode ser considerada um projeto com prazo não maior que um mês, que contempla a definição do que deve ser construído e um plano projetado e flexível para guiar a construção, o trabalho e o resultado do produto (peça 19, p. 8).*

*92. Embora incomum, uma **sprint** pode ser cancelada antes do seu **time-box** (tempo previsto) terminar. Porém, somente o **PO** tem a autoridade para cancelá-la, embora possa fazer isso sob influência das partes interessadas, da equipe de desenvolvimento ou do **Scrum Master**. O cancelamento ocorre se o seu objetivo se tornar obsoleto, ou seja, quando não fizer mais sentido sua execução às dadas circunstâncias (peça 19, p. 8-9).*

*Reunião de planejamento da **sprint***

*93. O trabalho a ser realizado na **sprint** é planejado na reunião de planejamento com o trabalho colaborativo de todo o time **Scrum**. Possui tempo máximo estabelecido e consiste em duas partes, cada uma ocupando a metade desse tempo (peça 19, p. 9).*

*94. Na primeira parte da reunião, define-se o que será entregue como resultado do incremento da **sprint**, sendo atribuição da equipe de desenvolvimento, e somente dela, selecionar os itens ordenados do **backlog** do produto previamente apresentado pelo PO que serão implementados (peça 19, p. 9).*

*95. Após a equipe de desenvolvimento escolher os itens que serão entregues, o time **Scrum** determina a meta da **sprint**, que consiste em um objetivo que será conhecido por meio da implementação do **backlog** do produto, fornecendo orientação relativa acerca da motivação do incremento (peça 19, p. 9-10).*

*96. Na segunda parte da reunião de planejamento, define-se como o trabalho necessário para entregar o incremento será realizado. Os itens de **backlog** do produto selecionados para a **sprint** junto com o plano de entrega desses itens é chamado de **backlog** da **sprint** (peça 19, p. 10).*

Reunião diária

*97. A reunião diária do **Scrum** é um evento de no máximo quinze minutos, destinado à equipe de desenvolvimento para sincronizar as atividades e criar um plano para as próximas 24 horas. É feita para inspecionar o trabalho desde a última reunião diária, e prever o trabalho que deverá ser feito antes da próxima reunião diária (peça 19, p. 10-11).*

98. As reuniões diárias melhoram a comunicação, eliminam outras reuniões, identificam e removem impedimentos para o desenvolvimento, destacam e promovem rápidas tomadas de decisão, e melhoram o nível de conhecimento da equipe de desenvolvimento (peça 19, p. 11).

*Revisão da **sprint***

*99. A revisão da **sprint** é uma reunião informal executada ao seu término para inspecionar o incremento e adaptar o **backlog** do produto, se necessário. Assim como os outros eventos, possui duração máxima pré-estabelecida. Nela, o incremento produzido é apresentado, destinando-se a motivar, obter comentários e promover a colaboração (peça 19, p. 11).*

*100. Durante a revisão da **sprint**, o time **Scrum** e as partes interessadas colaboram sobre o que foi executado. Com base nisso e em qualquer mudança no **backlog** do produto, os participantes colaboram nas próximas tarefas que precisam ser prontas (peça 19, p. 11).*

*101. Em suma, na reunião de revisão, o **PO** identifica o que ficou 'pronto' e o que não ficou 'pronto'; a equipe de desenvolvimento discute o que foi bem durante a **sprint**, quais problemas ocorreram e como foram resolvidos; a equipe de desenvolvimento demonstra o trabalho que está 'pronto' e responde as questões sobre o incremento; o **PO** discute o **backlog** do produto tal como está e projeta as prováveis datas de conclusão baseado no progresso corrente; e o grupo todo colabora sobre o que fazer a seguir, fornecendo valiosas entradas para a reunião de planejamento da próxima **sprint** (peça 19, p. 12).*

*102. O resultado da reunião de revisão da **sprint** é um **backlog** do produto revisado que define o provável **backlog** do produto para a próxima **sprint**. O **backlog** também pode ser ajustado completamente para atender novas oportunidades (peça 19, p. 12).*

*Retrospectiva da **sprint***

*103. A retrospectiva da **sprint** é uma oportunidade para o time **Scrum** revisar a si próprio e criar um plano de melhorias a serem aplicadas na próxima **sprint**. Ocorre depois da reunião de revisão e antes da reunião de planejamento da próxima **sprint**. É uma reunião com duração máxima pré-estabelecida (peça 19, p. 12).*

*104. Durante cada reunião de retrospectiva, o time **Scrum** planeja formas de aumentar a qualidade do produto, adaptando a definição de 'pronto', quando apropriado. Ao seu final, o time **Scrum** deverá ter identificado melhorias que serão implementadas na próxima **sprint** (peça 19, p. 12).*

*Artefatos do **Scrum***

*105. Os artefatos do **Scrum** representam o trabalho ou o valor de várias maneiras úteis para prover transparência e oportunidades para inspeção e adaptação. Os artefatos definidos no **Scrum** são especificamente projetados para maximizar a transparência das informações-chave necessárias para assegurar que o time **Scrum** tenha sucesso na entrega do incremento 'pronto' (peça 19, p. 12).*

***Backlog** do produto*

*106. O **backlog** do produto é uma lista ordenada ou priorizada de tudo que deve ser necessário no produto, e é uma origem única dos requisitos para qualquer mudança a ser feita nele. O **PO** é responsável pelo **backlog** do produto, incluindo seu conteúdo, disponibilidade e ordenação ou priorização (peça 19, p. 13).*

*107. Partindo-se da premissa de que os primeiros desenvolvimentos apenas estabelecem os requisitos inicialmente conhecidos e melhor entendidos, entende-se que o **backlog** do produto nunca está completo. Ao contrário, ele evolui tanto quanto o produto e o ambiente no qual ele será utilizado evoluem. Também é dinâmico, mudando constantemente para identificar o que o produto necessita para ser mais apropriado, competitivo e útil. Existe enquanto perdurar o produto (peça 19, p. 13).*

*108. O **backlog** do produto lista todas as características, funções, requisitos, melhorias e correções que formam as mudanças que devem ser feitas no produto em futuras versões, consistindo de um grupo de itens com atributos de descrição, ordem e estimativa. Geralmente, esses itens são ordenados por valor, risco, prioridade e necessidade. Os itens no topo da lista do **backlog** do produto determinam as atividades de desenvolvimento mais imediatas. Quanto maior sua posição no topo da lista, mais o item deve ser considerado e mais consenso existe em relação a ele e ao seu valor para o produto (peça 19, p. 13).*

*109. Uma vez que os itens do **backlog** do produto de ordem mais alta devem ser os primeiros a serem implementados, eles devem ser mais claros e detalhados que os itens de ordem mais baixa, possibilitando estimativas mais precisas. Os itens do **backlog** do produto que irão ocupar o desenvolvimento na próxima **sprint** são mais refinados, tendo sido decompostos de modo que todos os seus componentes possam ser 'prontos' dentro da **sprint** seguinte. Os itens do **backlog** do produto que podem ser implementados pela equipe de desenvolvimento são considerados como 'disponíveis' ou 'executáveis' para seleção durante a reunião de planejamento (peça 19, p. 13).*

*110. A preparação do **backlog** do produto (**grooming**) é o ato de adicionar detalhes, estimar e ordenar ou priorizar seus itens. É um processo contínuo exercido em tempo parcial e de forma colaborativa entre o **PO** e a equipe de desenvolvimento. Contudo, somente a equipe de desenvolvimento, responsável pela efetiva construção do produto, é que tem a atribuição de realizar as estimativas (peça 19, p. 14).*

Backlog** da **sprint

*111. O **backlog** da **sprint** é um conjunto de itens do **backlog** do produto selecionados para a **sprint**, adicionado do plano para entrega do incremento do produto a ser utilizado, intencionando o alcance do objetivo da **sprint**. É a previsão da equipe de desenvolvimento sobre qual funcionalidade estará no próximo incremento e do trabalho necessário para entregá-la (peça 19, p. 14).*

Incremento

*112. O incremento é a soma de todos os itens do **backlog** do produto concluídos durante a **sprint**, acrescido de todos os outros das **sprints** anteriores. Ao final da **sprint**, um novo incremento deve estar 'pronto', o que significa que deve estar na condição utilizável e atender a definição de 'pronto' do time **Scrum**. Este deve estar na condição utilizável independentemente do **Product Owner** decidir por liberá-lo ou não (peça 19, p. 15).*

*113. Além dos papéis, eventos e artefatos do **Scrum**, o conceito de 'pronto' também é explorado no Guia do **Scrum**.*

*114. Quando o item do **backlog** do produto ou um incremento é descrito como 'pronto', todos devem entender o que o 'pronto' significa. Embora possa variar significativamente entre diversos times **Scrum**, os integrantes devem ter um entendimento compartilhado do que significa o trabalho estar completo, assegurando a transparência. Essa é a 'Definição de Pronto' para o time, usada para avaliar se o incremento do produto está, de fato, concluído (peça 19, p. 14).*

*115. A mesma definição deve orientar a equipe de desenvolvimento na avaliação de quantos itens do **backlog** do produto podem ser selecionados na reunião de planejamento da **sprint** (peça 19, p. 14).*

*116. À medida que times **Scrum** amadurecem, é esperado que a definição de 'pronto' seja expandida para incluir critérios mais rigorosos que assegurem maior qualidade ao produto.*

*117. Por fim, cumpre registrar que, de acordo com o Guia do **Scrum**, seus papeis, artefatos, eventos e regras são imutáveis e, embora seja possível implementar somente partes da metodologia, o resultado não é **Scrum**, que existe somente na sua totalidade, funcionando bem como um container para outras técnicas, metodologias e práticas (peça 19, p. 16).*

3.2.2 eXtreme Programming (XP)

*118. Em 1996, Kent Beck, um dos signatários do Manifesto Ágil, se tornou gerente responsável pelo **Chrysler Comprehensive Compensation System**, projeto de desenvolvimento para substituição dos **softwares** que processavam a folha de pagamento da empresa **Chrysler**, também conhecido como C3. A partir de então, Kent começou a refinar os métodos que utilizava para desenvolver programas e passou a escrever o primeiro livro sobre a metodologia batizada por ele como **eXtreme Programming**, mais conhecida como **XP**.*

*119. O **software** que foi desenvolvido era um estudo de caso de adoção de programação orientada por objetos, e Kent Beck foi inicialmente convidado a fazer parte da equipe com objetivo de realizar melhorias de desempenho na aplicação. No entanto, ao assumir a liderança da equipe, introduziu práticas nas quais acreditava que trariam melhoria para o **software** em desenvolvimento. Assim nasceu o **XP**, como uma compilação das melhores práticas utilizadas no projeto C3.*

*120. Atualmente, várias dessas práticas evoluíram e o **XP** já pode ser adotado em situações que não guardam muita relação com o contexto no qual surgiu. Ao contrário de **Scrum**, que foca principalmente nas práticas relacionadas à gerência, **XP** dá mais atenção às tarefas de programação em si.*

*121. Apesar de ser considerado um conjunto de melhores práticas, **XP** traz, na sua proposta, também valores e princípios. Os valores são critérios gerais e abstratos, enquanto as práticas são*

claras e concretas. Os princípios, por sua vez, fazem a ligação entre os valores e as práticas. A seguir, os valores, princípios e práticas são apresentados sucintamente.

Valores

122. Valor 1: comunicação.

*122.1. Enquanto os clientes têm visão dos problemas que desejam solucionar, os desenvolvedores dominam as técnicas que influenciam a forma de resolver o problema apresentado pelo cliente. O resultado do **software** é tão bom quanto a capacidade de ambos se comunicarem.*

122.2. Existem diversas formas para se estabelecer essa comunicação, mas algumas se apresentam como melhores do que outras. Diálogos são mais eficazes que videoconferências que, por sua vez, são melhores que telefonemas, sendo esses mais expressivos que emails e assim sucessivamente. O diálogo presencial evita que problemas de má compreensão e ambiguidades comprometam negativamente o produto final.

123. Valor 2: coragem.

*123.1. Durante o desenvolvimento de um **software**, é natural que as funcionalidades inicialmente imaginadas pelo cliente e desenvolvidas pela equipe sofram alterações, seja porque os clientes imaginaram outras formas de resolver o problema, seja porque o time descobriu novas soluções. É um aprendizado natural, mas que, em metodologias tradicionais, envolve uma tendência da equipe se proteger de mudanças, buscando garantias de que os rumos mantenham-se inalterados.*

*123.2. Este valor do **XP** preza pela proteção. A equipe estabelece mecanismos para evitar que uma alteração futura seja um transtorno ou represente um risco grande ao projeto. Algumas práticas foram propostas com esse fim.*

*124. Valor 3: **feedback**.*

*124.1. É sabido que, quanto mais cedo um problema é descoberto, menos prejuízos ele pode causar e maiores são as chances de resolvê-lo de forma barata em um projeto de desenvolvimento de **software**. Por isso, **XP** incentiva que se encurte ao máximo a defasagem de tempo entre o momento em que uma ação é executada e que o seu resultado é observado. Assim, existe um incentivo ao constante **feedback**, mesmo que, pra isso, as entregas de novas funcionalidades sejam realizadas no menor prazo possível.*

125. Valor 4: respeito.

*125.1. Outro valor do **XP** é o respeito, não apenas pelos outros membros da equipe, mas também respeito a si próprio. Alterações que comprometam negativamente o trabalho de outros desenvolvedores não devem ser feitas. E cada desenvolvedor deve se comprometer entregando o produto com a maior qualidade possível.*

126. Valor 5: simplicidade.

*126.1. Simplicidade guarda relação com a máxima difundida pela Toyota a partir da década de 1950: não se deve produzir mais que o necessário. Concluiu-se que o Princípio de Pareto também se aplica ao desenvolvimento de **software**, onde 20% das funcionalidades costumam gerar 80% ou mais do benefício esperado. É um grande desperdício entregar funcionalidades que não agregam valor ao negócio.*

Princípios

*127. O princípio da auto-semelhança sugere que, quando equipes **XP** encontrarem soluções que funcionem em um contexto, também devem procurar adotá-las em outras situações, mesmo que em escalas diferentes. O princípio do benefício mútuo, por sua vez, prega que práticas que não beneficiem todos os envolvidos são capazes de destruir relacionamentos e criar mais dificuldades ainda nos projetos.*

*128. Outro ponto é a assunção de que, em se tratando de **software**, diferentes abordagens para um mesmo problema podem implicar em enormes diferenças no tempo e custo de implementação da solução. O princípio da diversidade ajuda a complementar as soluções e torná-las mais ricas.*

129. Quando duas abordagens para resolver um problema parecerem equivalentes, recomenda-se testar as duas e não ter medo de errar. O princípio da falha preceitua que com os erros advêm novos conhecimentos.

*130. Por outro lado, o princípio da economia estabelece que, ao se desenvolver **software**, é importante implementar as funcionalidades que puderem gerar maior retorno financeiro o mais cedo possível. Ao mesmo tempo, o princípio da fluidez estabelece que, ao invés de impor obstáculos, por meio de etapas bem definidas, deve-se permitir que o desenvolvedor aprenda sobre um requisito e avance rapidamente para a sua implementação. Para isso, ele fará um pouco de análise, de **design**, de teste, de implementação, e voltará a fazer um pouco mais de análise, teste, **design**, etc. Por seu turno, o princípio da melhoria preceitua que, não se deve buscar a perfeição em cada uma dessas etapas, mas sim aperfeiçoar esses e outros aspectos dos projetos, em um processo de melhoria contínua.*

*131. O princípio da oportunidade considera que problemas eventualmente encontrados devem ser vistos como oportunidades de aprendizado e mudança, levando a atitudes mais proveitosas para todos os envolvidos. Errar é algo natural em qualquer projeto e ocorre de tempos em tempos. Erros não devem ser temidos, a abrangência dos mesmos e o tempo necessário para descobri-los é que deve ser uma preocupação. Quando os erros são descobertos rapidamente e abrangem um escopo pequeno do trabalho, solucioná-los torna-se mais fácil. Por isso, segundo o princípio dos 'passos de bebê' ou pequenos passos, é melhor avançar um pouco de cada vez, ao invés de tentar grandes passos sem validar suas consequências. Todos esses princípios conduzem ao princípio da qualidade, que deve ser o foco durante todo o projeto de desenvolvimento do **software**. Existe uma crença de que alta qualidade significa gastos mais elevados. Não há dúvidas de que qualidade tenha preço, mas a falta dela tem um preço ainda maior.*

*132. Ainda com foco na qualidade, o princípio da redundância apresenta-se como outra maneira de garanti-la. Os problemas difíceis e críticos em desenvolvimento de **software** devem ser resolvidos de várias formas diferentes. Mesmo que uma solução falhe completamente, as outras irão prevenir um desastre. O custo da redundância é pago pela economia de não ter um desastre. O princípio da reflexão recomenda que as equipes não apenas façam seu trabalho, mas também pensem sobre como estão trabalhando e por que motivos estão-no fazendo. Devem analisar o porquê de terem resultado em sucesso ou fracasso. Não devem tentar esconder seus erros, mas expô-los e aprender com eles.*

133. Finalmente, o princípio da responsabilidade define que esta não pode ser atribuída, só pode ser aceita. Quando uma responsabilidade é atribuída a um indivíduo, somente ele pode decidir se a aceita ou não.

Práticas

*134. Em **XP**, as práticas distinguem-se em dois grupos: primárias e corolárias. As primeiras podem ser adotadas imediatamente, de forma segura, para melhorar o esforço de desenvolvimento de **software**. Por sua vez, as práticas corolárias são mais difíceis de serem implementadas antes de se adotarem as práticas primárias. Assim, devem ser usadas com maior cautela e com segurança de que a equipe e o ambiente encontram-se maduros para tal.*

*135. São práticas primárias: Ambiente Informativo, **Build** de Dez Minutos, Ciclo Semanal, Ciclo Trimestral, Desenvolvimento Orientado a Testes, **Design** Incremental, Equipe Integral, Folga, Histórias, Integração Contínua, Programação em Par, Sentar-se Junto e Trabalho Energizado.*

136. São práticas corolárias: Análise da Raiz do Problema, Base de Código Unificada, Código Coletivo, Código e Testes, Continuidade da Equipe, Contrato de Escopo Negociável, Envolvimento do Cliente Real, Equipes que Encolhem, Implantação Diária, Implantação Incremental e Pagar Por Uso.

3.2.3 Kanban

*137. **Kanban** é uma palavra japonesa e significa 'cartão visual'. Essa palavra é utilizada para descrever o sistema que a empresa Toyota utiliza desde a década de 1950 para controlar a linha de produção de seus veículos. Como mencionado anteriormente (item 51), o Sistema de Produção Toyota objetiva aumentar a eficiência da produção pela eliminação contínua de desperdícios.*

*138. A metodologia **Kanban** para desenvolvimento de **software** foi proposta por David J. Anderson e define um **framework** para melhoria incremental de processos e sistemas em organizações. A adoção do **Kanban** é ancorada na filosofia de que, para aperfeiçoar um processo, deve-se começar com o que se está fazendo agora, concordar em buscar mudanças incrementais e evolucionárias, além de respeitar o processo atual, com seus papeis, responsabilidades e cargos. Destaca-se também a necessidade de fomentar a postura de liderança em todos os níveis da organização.*

139. A metodologia não define um conjunto específico de passos ou de funções que deve ser seguido no processo de desenvolvimento. A adoção de um sistema de aperfeiçoamento baseado na metodologia começa com a organização dos passos e processos já utilizados, para posterior estímulo à melhoria contínua.

140. O aperfeiçoamento do sistema é obtido por meio da melhoria contínua e incremental no processo. Mudanças radicais não são bem vindas, pois apesar de parecerem mais efetivas, elas têm probabilidade de erro maior porque a organização tende a resistir. A idéia é estimular pequenos incrementos no processo a fim de se alcançar grandes melhorias no sistema.

141. Respeitar o processo corrente, os papeis, as responsabilidades e os títulos da organização dissipa o medo inicial de adoção de uma nova forma de trabalho, e a iniciativa de implantação da metodologia se estabelece.

*142. O **Kanban** é uma metodologia para impulsionar mudança, mas é pouco prescritivo, o que o diferencia de métodos como **XP** e **Scrum**. Não se define, de antemão, como trabalhar ou quais papéis devem ser adotados pela organização. Conforme declarado por Andersen, 'o **design** do sistema **Kanban** é um processo de pensamento; não uma cópia ou modelo de implementação de processo'. Também não é substituto para adoção de **Scrum** ou **XP**. A adoção de **Scrum** pode ser utilizada como ponto de partida para a utilização de **Kanban**, e este pode ser um catalisador da melhoria do processo adotado.*

*143. Tendo como base esses valores, a adoção de **Kanban** contempla as seguintes práticas:*

Visualizar o trabalho em andamento

144. A visualização do trabalho em andamento tem como objetivos manter o foco no 'todo', estimular a transparência no progresso das atividades e identificar desperdícios.

145. Para alcançar tais objetivos, deve-se entender o que está sendo feito atualmente; resistir à tentação de realizar mudanças inicialmente; mapear o fluxo de trabalho, identificando estágios e o tempo que se espera entre eles; e criar um quadro (eletrônico ou físico) com uma raia para cada estágio.

Limitar o trabalho em progresso

146. O objetivo dessa prática é identificar gargalos no fluxo, aumentando a produtividade. Nesse ponto, faz-se necessário esclarecer os seguintes conceitos:

*146.1. sistema **kanban** (k minúsculo): sistema de aperfeiçoamento de processos de trabalho baseado na metodologia **Kanban** (k maiúsculo);*

*146.2. tempo de ciclo: tempo que um item leva para passar pelo sistema **kanban**. Exemplo: quantas semanas uma história de usuário leva para ser entregue desde que foi identificada;*

*146.3. **Work In Progress (WIP)**: total de trabalho em progresso no sistema **kanban**. Exemplo: quantidade de histórias de usuário que estão atualmente em progresso;*

*146.4. produção (**throughput**): média do número de itens produzidos em um determinado período de tempo. Exemplo: duas histórias produzidas por semana;*

*146.5. tempo médio de ciclo: razão entre total de **WIP** e produção (**WIP**/produção). Para reduzir o tempo médio, pode-se aumentar a produção ou diminuir o **WIP**. Aumentar a produtividade da equipe não é o mais fácil, e por isso limitar o **WIP** aparece como solução para diminuir o tempo médio.*

*147. A definição do **WIP** de cada estágio deve seguir o bom senso e as particularidades da equipe, mas cinco podem ser usados como ponto de partida. Os limites iniciais são apenas palpites*

que são dados em momentos de pouca informação. À medida que se obtém informações sobre o sistema e que a forma de trabalho é aprimorada, os limites são ajustados.

148. Para evitar gargalos, o tamanho dos itens também é considerado, pois itens grandes bloqueiam recursos por muito tempo, enquanto itens pequenos fluem mais rapidamente pelo sistema. Quebrar histórias de usuários em partes pequenas – 'mínimo de funcionalidade vendável' – requer habilidade.

Explicitar as políticas que estão sendo seguidas

149. A importância que se dá à qualidade vem do alto custo de encontrar e corrigir defeitos em um item depois de sua entrega.

150. As políticas são representadas por padrões e listas de verificação para completar uma tarefa.

151. Para deixar explícitas, as políticas são acrescentadas ao quadro, observando que estágios podem ser inteiramente dedicados à garantia da qualidade.

152. O objetivo final dessa prática é a melhoria na qualidade do produto.

Medir e gerenciar o fluxo

*153. O sistema de entrega de **software** tem capacidade limitada. Pressionar além da capacidade tem como consequência a queda da qualidade.*

*154. Estimativas de capacidade realizadas no início de um projeto são palpites baseados em projetos anteriores. Em um sistema **kanban**, planos são usados como guias e não como critério de sucesso.*

155. Para obter um sistema de entrega estável e previsível e suportar a tomada de decisão a respeito de prazos, dependências, pessoal e escopo, a medição do progresso é importante desde o início. Porém, toda medição realizada é atrelada a um objetivo para evitar medições desnecessárias.

156. São vários os tipos de medições que podem ocorrer, mas a maioria deles diz respeito à quantidade de trabalho realizado ou que falta ser realizado, tempo de duração do ciclo, índice de defeitos do produto e itens bloqueados em estágios no sistema. Gráficos com as medições são expostos junto ao quadro.

157. A fila de entrada dos itens a serem tratados deve estar sempre ordenada de acordo com a prioridade, a fim de que as pessoas possam puxar os itens que estejam no topo. Para fazer a priorização, são levados em consideração diversos fatores: riscos e incertezas; necessidades básicas, como infraestrutura; manutenção de tamanho dos itens equilibrados para um fluxo constante; tipos de história também equilibrados, para manutenção do fluxo de entrega de valor; e dependências entre os itens. Definir um item como prioritário significa abrir mão de priorizar outro. As escolhas devem ser conscientes.

158. Para buscar a melhoria do sistema, a análise se apóia em filtros de decisão. Os filtros ajudam a manter o foco na melhoria do fluxo como um todo e não em atividades individuais.

Incentivar a melhoria contínua

*159. A adoção das práticas anteriores já permite a identificação de oportunidades de aperfeiçoamento, mas o **Kanban** atinge processo contínuo de melhoria com a adoção de eventos de retrospectiva de forma cadenciada e regular. Com as reuniões de retrospectiva, o time concentra-se no fluxo e identifica oportunidades de mudanças estruturais maiores.*

*160. Na Figura 1, observa-se imagem de um quadro físico extraído do livro '**Kanban** em 10 passos'. O sistema apresentado é utilizado para aperfeiçoamento de um processo de desenvolvimento de **software**. Os estágios identificados estão dispostos nas raias: **Inbox**, **Specification**, **Ready for Development**, **Development**, **Code Review**, **Teste Locally**, **Test on PreProduction**, **Ready for Release**. Os números em vermelho, logo abaixo do nome do estágio, representam o limite de **WIP** para aquele estágio. Na parte abaixo do quadro, estão as políticas de cada estágio. Os itens estão distribuídos em*

cartões pelas raias. Assim, os gráficos acima do quadro deixam transparentes as medições realizadas.

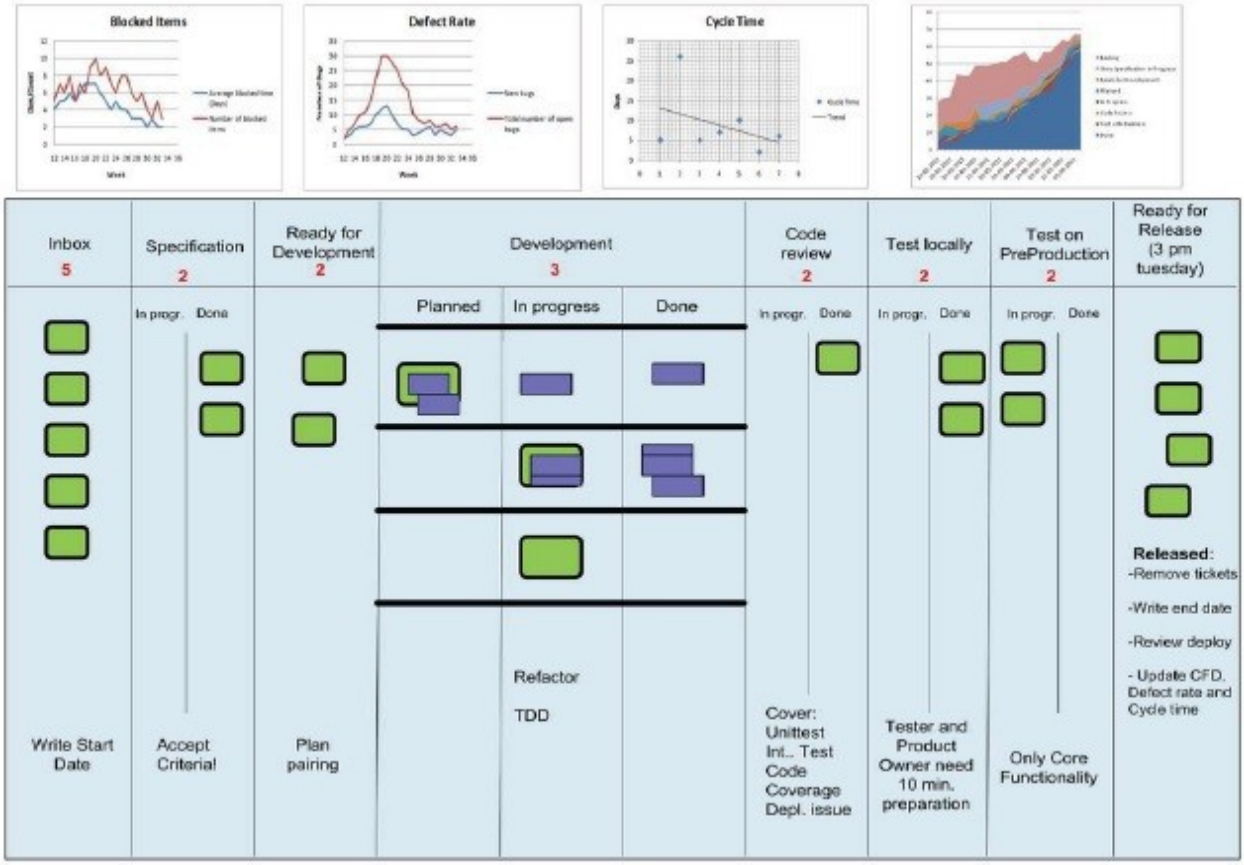

*Figura 1: Exemplo de quadro **kanban** para processo de desenvolvimento de **software**.*

*161. O livro **Kanban**, de David J. Anderson, descreve a abordagem **Kanban** para sistemas de aperfeiçoamento de processos de desenvolvimento de **software**.*

3.3. Comparação entre metodologias ágeis e metodologias tradicionais

162. Comparando-se as metodologias tradicionais com a essência que fundamenta os métodos ágeis, é possível destacar quais as diferenças e semelhanças entre as duas metodologias.

Processo adaptativo x processo preditivo

163. A maioria das metodologias tradicionais se apresenta como processos preditivos, onde várias atividades, tarefas, papéis e artefatos são definidos. Apesar de serem vendidas como processos que devem ser adaptados, identificar quais desses passos são dispensáveis dentro de um determinado projeto quase sempre se mostra uma tarefa complexa.

164. Os métodos ágeis, por outro lado, apresentam-se como ferramentas e métodos adaptativos, com poucos procedimentos e papeis, reforçando seu foco em melhores práticas, princípios e valores. Para que um projeto adote uma metodologia ágil, é necessário trazer e adaptar os valores apresentados para o contexto do projeto. Dessa maneira, enquanto as metodologias tradicionais são chamadas de preditivas, as ágeis são consideradas adaptativas.

Comunicação direta

*165. As metodologias tradicionais são também chamadas de pesadas ou orientadas à documentação. Além de detalharem as atividades que se deve executar durante o desenvolvimento do **software**, também incentivam a confecção de número considerável de documentos, como modelos, diagramas e especificações. Desta forma, a principal maneira de comunicação entre as pessoas é baseada em documentos formais.*

166. Essa abordagem não é dispensada pelas metodologias ágeis, mas há uma valorização maior na interação direta entre as pessoas de uma equipe a fim de melhorar a transmissão e disseminação de conhecimento entre os indivíduos.

Aceitação da mudança

*167. Os métodos ágeis introduziram no desenvolvimento de **software** alguns recursos que permitem uma maior liberdade para experimentação. Ao invés de exigir do cliente que ele saiba exatamente qual o comportamento ideal do **software** a ser desenvolvido logo no início, como no modelo cascata, mecanismos foram estabelecidos para que suas funcionalidades sejam revistas, caso se descubra uma maneira mais adequada para construí-lo, assemelhando-se, em certa medida, ao modelo em espiral. O processo de experimentação e adaptação incentivado pelos métodos ágeis tende a produzir programas mais adequados às necessidades dos usuários.*

Incentivo a ciclos curtos e entregas constantes

*168. Apesar de terem sido propostos como iterativos e incrementais, as metodologias baseadas no processo Unificado (**UP**) nem sempre atingiram tal objetivo.*

*169. Por outro lado, os métodos ágeis sugerem ciclos de desenvolvimento bastante curtos, quase nunca superiores a um mês, incentivando ao máximo que **softwares** sejam desenvolvidos de forma iterativa e incremental. Os ciclos curtos também possibilitam **feedback** constante, o que reflete em maior qualidade para o processo e para o produto.*

*170. Ainda com objetivo de experimentar e obter **feedback**, os métodos ágeis ressaltam a importância de liberar o **software** em ambiente de produção o mais rápido possível, de preferência ao final de cada ciclo de desenvolvimento, mesmo que poucas funcionalidades tenham sido implementadas. A certeza de que a evolução do sistema está acontecendo de maneira adequada só é constatada com o retorno positivo dos usuários reais do produto.*

4. Os valores ágeis e os princípios da Administração Pública

*171. A essência dos métodos ágeis para desenvolvimento de **software** baseia-se nos valores fundamentais contidos no Manifesto Ágil. Confrontando-se esses valores com os princípios da Administração Pública consignados em dispositivos legais, a exemplo da Constituição Federal e da Lei de Licitações e Contratos, é possível avaliar o alinhamento da essência ágil com o ordenamento jurídico vigente sob a perspectiva da terceirização pelos órgãos públicos federais, objeto principal desta fiscalização. Importa observar que a interpretação desses valores é subjetiva, e que a exposta a seguir corresponde à ótica de controle desta unidade técnica.*

Indivíduos e interação entre eles, mais que processos e ferramentas

171.1. A maior valorização de 'indivíduos e interação entre eles' em detrimento de 'processos e ferramentas' constitui pilar basilar das equipes de desenvolvimento ágil. Contudo, pode entrar em atrito com o princípio da eficiência por possibilitar que os processos da instituição possam ser relegados. Adicionalmente, uma vez que o princípio em epígrafe valoriza as pessoas, a substituição de membros da equipe durante a execução de um projeto contrasta com o princípio da eficiência, haja vista que pode acarretar prejuízos à produtividade da equipe ágil.

171.2. Além disso, o destaque para os indivíduos e suas interações pode contribuir para a construção de uma relação de pessoalidade entre os funcionários da contratada e os gestores da instituição contratante, prática condenada pelo Tribunal Superior do Trabalho (TST) em seu Enunciado 331.

***Software** em funcionamento, mais que documentação abrangente*

*171.3. A maior valorização de '**software** em funcionamento' em detrimento de 'documentação abrangente', embora possa vir ao encontro do princípio da eficiência, possibilitando a incorporação, de forma mais célere, de sistemas informatizados necessários à missão da Administração Pública, paradoxalmente também o fere. Menosprezar a adequada documentação do **software** contratado pode ocasionar problemas para a sua manutenibilidade e, por consequência, a continuidade de seufuncionamento de modo adequado.*

*171.4. Para mitigar esse risco, um conjunto mínimo de artefatos – entre eles documentos essenciais à sustentação dos **softwares** – deve ser exigido no instrumento convocatório. Para auxiliar a tarefa de definição dos artefatos que devem acompanhar o sistema entregue pela contratada a cada*

*iteração, a instituição pode utilizar sua Metodologia de Desenvolvimento de Sistemas (MDS) ou seu processo de **software**, adaptado e construído segundo sua realidade específica.*

Colaboração com o cliente, mais que negociação de contratos

171.5. A maior valorização da 'colaboração com o cliente' em prejuízo da 'negociação de contratos', em primeira análise, entra em atrito com o princípio da vinculação ao instrumento convocatório, uma vez que pode fazer com que a contratada execute serviços não cobertos pelo contrato, ocasionando enriquecimento sem causa da Administração. Em contratações públicas, é imperativo que essa prática seja abolida.

Responder a mudanças, mais que seguir um plano

171.6. A valorização de 'responder a mudanças' em detrimento de 'seguir um plano' contrasta, de imediato, com o princípio fundamental do planejamento, o qual é estampado no inciso I do art. 6º do Decreto-Lei 200/1967, e pode ser conflitante com o princípio da economicidade, insculpido no art. 70 da Carta Magna de 1988.

*171.7. Em tese, fere o princípio do planejamento por permitir que a tarefa de desenvolvimento de **software** se afaste das diretrizes e metas inicialmente estipuladas ou, até mesmo, em extremada circunstância, a elaboração do planejamento seja desprezada. Adicionalmente, mudanças constantes podem revelar esforço de planejamento inadequado para a definição dos requisitos do **software** contratado.*

171.8. Por sua vez, a maior valorização da resposta a mudanças opõe-se ao princípio da economicidade por exigir da empresa contratada retrabalho para o ajuste às mudanças, podendo acarretar novos desembolsos ao erário.

*171.9. Entretanto, o valor ágil em epígrafe também vai ao encontro do princípio constitucional da eficiência ao permitir, por exemplo, a incorporação de novos requisitos oriundos de necessidades prementes no **software** em desenvolvimento.*

*172. Embora em primeira análise possa transparecer que a utilização de métodos ágeis em contratações públicas para desenvolvimento de **software** seja impossibilitada pelo conflito existente entre os seus valores e os princípios da Administração Pública, a análise dos editais e contratos das instituições públicas federais visitadas demonstra que, na prática, é possível alinhar a sua utilização aos preceitos legais que regem a esfera pública.*

5. Terceirização de desenvolvimento de software com métodos ágeis nas instituições da Administração Pública Federal visitadas

*173. A equipe de fiscalização visitou cinco órgãos que possuíam contratos de desenvolvimento de **software** utilizando métodos ágeis para sua construção, consistindo no objeto de interesse para o presente levantamento. Ressalte-se que das oito instituições inicialmente selecionadas, apenas cinco possuíam contratos de interesse.*

174. A seguir, são apresentados relatos acerca das contratações analisadas quanto a questões pertinentes à fiscalização, como métrica utilizada, forma de gerir as demandas e a execução dos serviços e níveis de serviço estabelecidos. Em adição, é relatado o motivo pelo qual algumas das instituições visitadas não apresentaram interesse ao levantamento.

5.1. Visão geral dos contratos de terceirização

*175. Dos órgãos visitados que possuíam objeto de interesse para a presente fiscalização, cabe observar que alguns possuem boa estrutura interna de TI, como o Tribunal Superior do Trabalho (TST), Banco Central do Brasil (Bacen) e Supremo Tribunal Federal (STF). Essas três instituições possuem equipes de servidores do próprio quadro que atuam no desenvolvimento de **software** utilizando métodos ágeis.*

176. Por outro lado, o Instituto do Patrimônio Histórico e Artístico Nacional (Iphan) possui grande carência de profissionais de TI em seu quadro. Todo o desenvolvimento de novos sistemas de informação é executado por empresas terceirizadas. Já o Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira (Inep), sob a perspectiva de pessoal da área de TI, encontra-se em situação momentaneamente mais confortável do que o Iphan. A área de TI do Inep, quando da visita

da equipe de fiscalização, contava com cinco servidores do quadro, além de trinta e sete profissionais com Contratos Temporários da União (CTU). Contudo, alguns desses profissionais deverão deixar o Instituto no próximo ano em virtude do término da vigência de seus contratos temporários de trabalho, não havendo previsão de que venham a ser substituídos.

Tribunal Superior do Trabalho (TST)

*177. O TST, após adquirir experiência interna em desenvolvimento de **software** utilizando **Scrum**, publicou, em 2012, o edital do Pregão Eletrônico 146/2012, o qual teve por objeto a contratação de Horas de Serviço Técnico (HST) de desenvolvimento de sistemas para os ambientes do Tribunal (peça 20, p. 1).*

178. Os serviços abarcados pelo objeto do pregão em comento constituíram-se de iniciação de sustentação de sistema, sustentação programada, sustentação emergencial e implantação, os quais deveriam ser cotados separadamente pelas licitantes (peça 20, p. 2).

*179. A sustentação programada englobou manutenções evolutivas e corretivas, cujo modelo de execução dos serviços foi descrito no Anexo 2 (Modelo de Sustentação de Sistemas do TST) do termo de referência do edital (peça 20, p. 49-63). Esse modelo baseou-se fundamentalmente no **framework Scrum**. Entretanto, o TST incluiu em seu detalhamento diferenciações do modelo original, tais como as previstas nos subitens 3.6 e 9.1 (peça 20, p. 50 e 56).*

*180. Segundo o subitem 3.6, cada **sprint** pode permitir o agrupamento de manutenções de vários sistemas diferentes. Como consequência, um time de desenvolvimento **Scrum** provavelmente deverá interagir com mais de um **Product Owner** (**PO**) e coordenar atividades distintas de sistemas diversos, o que pode trazer ineficiência ao processo de desenvolvimento.*

*181. O subitem 9.1 define o papel do **PO** e, ao contrário do preceituado no **framework Scrum**, prevê que este poderá ser desempenhado por mais de uma pessoa. Essa característica do modelo desenhado pelo TST pode trazer conflitos na definição da visão do produto, na gerência/priorização dos requisitos e na aceitação do produto entregue, acarretando ineficiência ao processo.*

*182. O Contrato 146/2012, decorrente do Pregão Eletrônico 146/2012, foi celebrado com a empresa **Squadra** Tecnologia em **Software** Ltda., em 31/12/2012, ao custo de R$ 1.572.740,00. A empresa deve executar os serviços demandados em suas próprias dependências, sendo que, eventualmente, atividades como reuniões de ponto de controle, definição de requisitos, entrega e demonstração dos produtos das ordens de serviço devem ser feitas nas dependências do TST (peça 20, p. 26).*

183. Quando da visita da equipe de fiscalização ao TST, foi verificado que nenhuma ordem de serviço ainda havia sido emitida. Tal fato impediu a coleta de informações acerca da experiência do TST com a gestão contratual.

Banco Central do Brasil (Bacen)

*184. O Bacen, assim como o TST, inicialmente introduziu os conceitos das metodologias ágeis para desenvolvimento de sistemas em suas equipes internas. Em 2012, por meio do Pregão Eletrônico Demap 7/2012, promoveu licitação para a contratação de serviços técnicos de tecnologia da informação para desenvolvimento, documentação e manutenção de sistemas de informação, dimensionados por meio de pontos de função, em regime de fábrica de **software** (peça 21, p. 1).*

185. Os serviços de desenvolvimento e manutenção dos sistemas de informação do Bacen seguem o Processo de Desenvolvimento Ágil do Bacen (PDS-Ágil) elaborado pela própria instituição, cujo fluxo de trabalho é apresentado no diagrama constante em documento específico que descreve a metodologia (peça 27, p. 3).

*186. A metodologia concebida pelo Bacen é fortemente influenciada pelo **Scrum**, incorporando conceitos e práticas desse framework, como **product backlog**, **Product Owner** e reuniões diárias; e pelo **XP**, a exemplo de simplicidade da solução implementada, integração contínua, programação em par, refatoração, desenvolvimento orientado por testes (**Test Driven Development – TDD**) e desenvolvimento orientado por testes de aceitação (**Acceptance Test Driven Development – ATDD**).*

187. Embora o PDS-Ágil esteja sendo empregado pela contratada para fornecer os serviços demandados pelo Bacen, observa-se da definição de escopo na versão atualmente utilizada que essa metodologia de desenvolvimento destina-se ao uso interno e, havendo subcontratação, pode ser aplicada às atividades não subcontratadas e a todos os produtos gerados (peça 27, p. 10, subitem 3.2). Entretanto, o PDS-Ágil não especifica quais atividades podem ser objeto de subcontratação. Em adição, os papéis nela definidos não vislumbram participação de atores externos ao Bacen (peça 27, p. 10-12).

188. A empresa vencedora do Pregão Eletrônico Demap 7/2012 foi a Politec Tecnologia da Informação S.A., que celebrou o Contrato Bacen/Deinf 50.412/2012 em 22/4/2012, com vigência compreendendo o período entre 23/4/2012 e 22/4/2013, ao custo de R$ 7.605.511,20 (peça 21, 143). Os serviços são prestados nas dependências da contratada, de modo remoto, podendo ser executados, a critério do Bacen, os serviços de projeto de construção, total ou parcialmente, em suas próprias dependências em Brasília (peça 21, p. 23).

189. Em 22/4/2013, foi celebrado o primeiro termo aditivo do referido ajuste visando prorrogar a vigência contratual até 22/4/2014 e reajustar o valor unitário do ponto de função, elevando o valor estimado da contratação para R$ 8.085.254,40 (peça 21, p. 144).

Instituto do Patrimônio Histórico e Artístico Nacional (Iphan)

190. O Iphan teve sua primeira experiência com métodos ágeis por meio do Pregão Eletrônico 12/2011. Esse certame teve por objeto o desenvolvimento do Sistema Integrado de Conhecimento e Gestão (SICG), estimado em dois mil pontos de função (peça 22, p. 5-6). Na ocasião, sagrou-se vencedora a empresa EGL Engenharia Ltda., a qual celebrou o Contrato 28/2011 em 2/12/2011, com vigência estabelecida até 1/9/2013 e ao custo de R$ 990.000,00 (peça 22, 59). Segundo o contrato firmado, os serviços constantes de seu objeto devem ser executados nas instalações da contratada, sendo que algumas reuniões de controle inerentes à metodologia de gestão do projeto do Iphan devem ocorrer nas dependências do Iphan (peça 22, p. 52).

*191. A forma de execução dos serviços, exposta no subitem 7.4 do termo de referência do citado pregão, previu o desenvolvimento do projeto em ciclos e a realização de cerimônias bem definidas, assemelhando-se ao **framework Scrum** (peça 22, p. 29-33).*

*192. À época da visita da equipe de fiscalização ao Iphan, o órgão estava elaborando edital para a contratação de fábrica de **software** para o desenvolvimento e manutenção de seus sistemas (peça 27, 470-541). Segundo a minuta do termo de referência, esses serviços devem ser executados em conformidade com a Metodologia Iphan de Gestão de Demandas de Desenvolvimento Ágil de **Softwares** (Midas – peça 27, p. 476). O modelo de gestão estabelecido na Midas é baseado no **Scrum**, com poucas alterações (peça 22, p. 68).*

Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira (Inep)

*193. O Inep, diferentemente das demais instituições visitadas, atualmente encontra-se em seu segundo contrato de fábrica de **software** para desenvolvimento de sistemas com uso de métodos ágeis.*

*194. O Contrato 1/2011, decorrente do Pregão Eletrônico 11/2010, foi a primeira iniciativa do Instituto de contratação de desenvolvimento de sistemas de informação com métodos ágeis. Teve por objeto a contratação de fábrica de **software** para a prestação de serviços técnicos de tecnologia da informação, compreendendo desenvolvimento e manutenção de sistemas de informação ao custo de R$ 6.984.000,00 (peça 23, p. 3 e peça 23, 383). Esses serviços deveriam ser executados segundo a orientação da Metodologia de Gestão e Desenvolvimento de Sistemas do Inep (MGDS), a qual é baseada na combinação de práticas advindas do **XP** e do **Scrum** (peça 23, p. 89).*

*195. A empresa **Squadra** Tecnologia em **Software** Ltda. sagrou-se a vencedora do certame. Porém, segundo relatado na reunião com representantes da área de TI do Inep, essa experiência não produziu os resultados esperados. O principal motivo para o fracasso da iniciativa foi a disponibilização de equipe composta basicamente de analistas juniores e o seu desconhecimento em desenvolvimento de sistemas utilizando métodos ágeis. Ao final do contrato, a empresa **Squadra** optou*

por não renovar a vigência contratual, alegando que o preço cotado para o contrato não serviria para cobrir seus custos.

196. A segunda experiência do Inep, atualmente em execução, decorre do Pregão Eletrônico 14/2012 (peça 24, 1-498), realizado em substituição ao contrato resultante do Pregão Eletrônico 11/2010. Esse segundo pregão teve como vencedora a empresa Cast Informática S.A., a qual celebrou, em 10/9/2012, o Contrato 33/2012, com período de vigência compreendido entre 10/9/2012 a 9/9/2013 e ao custo anual de R$ 12.000.000,00 (peça 24, 499). O objeto do novo contrato manteve os mesmos serviços abarcados pelo Contrato 1/2011 (peça 24, p. 3), também devendo ser executado nas instalações do Inep de acordo com a MGDS do Inep (peça 24, p. 40).

197. Conforme relatado pelos representantes do Inep em reunião com os membros da equipe desta fiscalização, esse segundo contrato, após o período de adaptação da empresa contratualmente estabelecido em noventa dias, tem sido executado a contento, atendendo às necessidades do Instituto.

Supremo Tribunal Federal (STF)

*198. Inicialmente, o STF implantou o desenvolvimento de sistemas utilizando metodologia ágil, baseada no **Scrum**, em suas equipes internas. Em 2012, lançou o edital do Pregão Eletrônico 84/2012, cujo objeto consistiu na contratação de empresa para prestação de serviços de desenvolvimento ágil de soluções de **software** (peça 25, p. 2). Sagrou-se vencedora do certame a empresa **Basis** Tecnologia da Informação S.A., celebrando o Contrato 27/2013 em 6/5/2013, ao custo anual de R$ 2.295.000,00, devendo prestar os serviços nas instalações e com recursos de infraestrutura tecnológica do STF (peça 25, p. 73 e peça 25, p. 33).*

*199. A forma de execução dos serviços definida no item 9 (Gerenciamento da Contratação) do termo de referência do edital do Pregão Eletrônico 84/2012 baseou-se no **framework Scrum** (peça 25, p. 27-42). Quando realizada a visita ao STF, o órgão ainda não tinha iniciado a execução contratual, impedindo a equipe de fiscalização de colher de informações relativa à sua experiência.*

Departamento de Informática do Sistema Único de Saúde (Datasus)

*200. Em abril de 2013, época da visita da equipe de fiscalização ao Datasus, verificou-se que o órgão possuía dois contratos vigentes relativos ao desenvolvimento de sistemas. Um deles, celebrado com a empresa CTIS Tecnologia S.A. destinava-se ao levantamento de requisitos, enquanto o outro, celebrado com a empresa **Cast** Informática S.A., tinha como objetivo a implementação do sistema especificado pela primeira. Quando necessário, de acordo com um dos responsáveis do órgão pelo serviço de desenvolvimento do Datasus, o segundo contrato era utilizado para o desenvolvimento integral de sistemas por meio de métodos ágeis, embora não houvesse previsão no instrumento convocatório para que a empresa assim procedesse. Por tal motivo, os contratos e editais originários não foram solicitados.*

201. Em adição, verificou-se que, como o contrato com a empresa CTIS estaria próximo do término de sua vigência, o Datasus realizou o Pregão Eletrônico 19/2013 com o intuito de contratar empresas especializadas para prestação de serviços técnicos de desenvolvimento e manutenção de sistemas de informação estimados em 130.000 pontos de função e contagem de outros 210.000 (peça 26, p. 1).

Empresa Brasileira de Serviços Hospitalares (EBSERH)

202. Quando a equipe de fiscalização visitou a EBSERH, foi apresentada a metodologia adotada pelo Hospital de Clínicas de Porto Alegre para sustentação do Aplicativo de Gestão para Hospitais Universitários (AGHU), não havendo tempo hábil para a apresentação da metodologia utilizada no desenvolvimento de sistemas, objeto do presente levantamento. Por tal motivo, a equipe de fiscalização não solicitou maiores informações acerca dos instrumentos convocatório e contratual.

Serviço Federal de Processamento de Dados (Serpro)

*203. A visita da equipe de fiscalização ao Serpro serviu para coleta de informações referentes à utilização de métodos ágeis pelo lado do fornecedor. O Serpro desenvolveu o sistema Novo Siafi para a Secretaria do Tesouro Nacional (STN) utilizando o **framework Scrum**. Uma tentativa*

*fracassada usando metodologia tradicional, baseada no **Unified Process (UP)**, antecedeu essa experiência.*

*204. Na primeira tentativa, o Serpro basicamente produziu documentações relativas ao levantamento de requisitos, sem, contudo, entregar o sistema contratado. Devido ao fracasso da metodologia empregada, o Serpro propôs à STN a execução do serviço utilizando métodos ágeis, o que foi por ela aceito. Grande parte dos requisitos já especificados na tentativa anterior foi utilizada na construção do Novo Siafi, diminuindo o esforço de levantamento de requisitos e construção do **backlog** do produto.*

205. A seguir, é apresentado quadro que sintetiza as informações acerca das contratações identificadas pela equipe de fiscalização.

Órgão	***Pregão***	***Tipo do objeto da contratação***	***Empresa contratada***	***Base do framework de gerência utilizado***	***Interessa ao levantamento***
TST	*Pregão Eletrônico 146/2012*	*Fábrica de **software***	***Squadra** Tecnologia em **Software** Ltda.*	***Scrum***	*Sim*
Bacen	*Pregão Eletrônico Demap 7/2012*	*Fábrica de **software***	*Politec Tecnologia da Informação S.A.*	***Scrum***	*Sim*
Iphan	*Pregão Eletrônico 2/2011*	*Projeto*	*EGL Engenharia Ltda.*	***Scrum***	*Sim*
	TR em elaboração	*Fábrica de **software***	*Não se aplica*	***Scrum***	*Sim*
Inep	*Pregão Eletrônico 1/2010*	*Fábrica de **software***	***Squadra** Tecnologia em **Software** Ltda.*	***Scrum***	*Sim*
	Pregão Eletrônico 14/2012	*Fábrica de **software***	*Cast Informática S.A.*	***Scrum***	*Sim*
STF	*Pregão Eletrônico 84/2012*	*Fábrica de **software***	*Basis Tecnologia da Informação S.A.*	***Scrum***	*Sim*
Datasus	*Não obtido*	*Fábrica de **software** – levantamento de requisitos*	*CTIS Tecnologia S.A.*	*Metodologia tradicional*	*Não*
	Não obtido	*Fábrica de **software** - construção*	*Cast Informática S.A.*	*Metodologia tradicional*	*Não*
	Pregão Eletrônico 19/2013	*Fábrica de **software***	*CTIS Tecnologia S.A.*	***Scrum***	*Não*
EBSERH	*Informação não obtida*	*Fábrica de **software***	*Informação não obtida*	*Não obtido*	*Não*
Serpro	*Dispensa de licitação*	*Projeto*	*Não se aplica (Serpro é fornecedor)*	*Scrum*	*Não*

Quadro 1 – Resumo das contratações identificadas pela equipe de fiscalização

5.2. Métricas de tamanho e esforço

206. Da análise dos editais, contratos e extratos de contratos obtidos das instituições (de interesse a este trabalho) visitadas (peças 20 a 25), verifica-se que, à exceção do TST, todas elas

utilizaram pontos de função para dimensionar e pagar os serviços contratados, conforme apresentado na tabela seguinte.

Órgão	Contrato	Métrica	Quantitativo Anual	Valor Unitário (R$)	Valor Total (R$)
TST	146/2012	Horas de serviço técnico (HST)	1.060	49,00	1.572.740,00
			24.000	49,00	
			4.200	62,38	
			1.380	60,00	
Bacen	50.412/2012	Pontos de função	16.560	459,27	7.605.511,20
	1º Aditivo			488,24	8.085.254,40
Iphan	28/2011	Pontos de função	2.000	495,00	990.000,00
	Em planejamento	Pontos de função	5.000	-	-
Inep	1/2011	Pontos de função	20.000	349,20	6.984.000,00
	33/2012		20.000	625,00	12.500.000,00
STF	27/2013	Pontos de função	5.000	459,00	2.295.000,00

Quadro 2 – Métricas utilizadas nos contratos das instituições visitadas

207. No edital do Pregão Eletrônico 146/2012, o TST utilizou horas de serviço técnico (HST) para mensurar o esforço e remunerar a contratada pelos serviços prestados, cotando em separado valores para cada um dos serviços estabelecidos no edital, quais sejam iniciar sustentação de sistema, sustentação programada, sustentação emergencial e implantação (peça 20 – Edital TST, p. 2-4).

5.3. Planejamento, gestão de demandas, aceitação do produto e forma de pagamento

*208. A maior parte dos instrumentos convocatórios analisados contém formas de planejamento, gestão de demandas, aceitação do produto e pagamento semelhantes. A demanda para construção do produto é precedida pelo planejamento do produto, o qual pode ser feito apenas pela instituição contratante ou em conjunto, entre essa e a empresa contratada. Além do planejamento do produto em si, algumas instituições também fazem o planejamento das funcionalidades que serão implementadas no próximo ciclo, iteração ou **sprint**, atividade preceituada no **Scrum**.*

*209. A equipe de fiscalização também observou que as instituições visitadas emitem uma ordem de serviço por ciclo, iteração ou **sprint**, ou por **release** de **software**, sendo mais comum o primeiro caso.*

*210. Quanto à aceitação do produto entregue pela contratada, embora no **framework Scrum** seja preceituado que ocorra na reunião de revisão da **sprint**, essa prática não é executada nos contratos estudados, até mesmo por impedimento normativo, como disciplinado no art. 73 da Lei 8.666/1993. Nessa ocasião, algumas instituições apenas verificam se os artefatos exigidos foram entregues, caracterizando o recebimento provisório.*

*211. Acerca da forma de pagamento da contratada, constatou-se que algumas instituições remuneram os serviços de planejamento quando realizados, enquanto outras remuneram apenas os serviços de construção do **software**.*

*212. A entrega adiantada e contínua de **software**, conforme postulado nos princípios dos métodos ágeis, foi observada em algumas das instituições visitadas. Para alcançar esse objetivo, elas paralelizam as atividades de preparação, execução e homologação, isto é: em um dado período de tempo, enquanto a empresa contratada executa a construção do **software** em um ciclo, iteração ou **sprint**, a contratante prepara os itens do **backlog** do produto que serão implementados no próximo ciclo e homologa o produto entregue no ciclo anterior.*

213. A forma de planejamento, gestão de demandas, aceitação do produto e forma de pagamento pelos serviços executados nos contratos examinados são apresentados a seguir.

Tribunal Superior do Trabalho (TST)

*214. No TST, a execução do serviço de sustentação programada, um dos objetos do Contrato 146/2012, é baseada no **Scrum**, sendo realizada em **sprints**, cujo acompanhamento é dado pela*

*atualização de quadros **kanban** (peça 20, p. 50 e 61). Para que um sistema do Tribunal possa sofrer esse tipo de intervenção pela empresa contratada, inicialmente ele passa pelo processo de iniciação da sustentação, no qual a contratada gera seus respectivos manual de produção e documento de arquitetura (peça 20, p. 51). A partir desse momento, a sustentação programada no sistema dá-se por meio de **sprints**.*

*215. Cada **sprint** do serviço de sustentação programada é precedida por uma Solicitação de Sustentação Programada, a qual é formalizada por ordem de serviço (OS). O real objetivo desse tipo de solicitação é planejar a **sprint**, gerando como artefato o Plano da **Sprint** (peça 20, p. 52). Esse artefato contém, entre outros elementos, as histórias de usuário (**backlog** da **sprint**) e os respectivos critérios de aceite, datas de início e fim da **sprint** e data, hora e local da reunião de apresentação dos seus resultados (peça 20, p. 57-58). O TST, a seu critério, identifica histórias de usuários essenciais, que devem ser implementadas em cada **sprint** (peça 20, p. 26). Caso uma dessas histórias não seja entregue ao final da **sprint**, ela é dada como fracassada. Cada história de usuário é medida em pontos de função com vistas à apuração da produção e da remuneração da contratada (peça 20, p. 53).*

*216. Com o Plano da **Sprint** preparado, inicia-se a sustentação programada propriamente dita, na qual as funcionalidades especificadas no planejamento devem ser codificadas, testadas, documentadas e apresentadas ao TST (peça 20, p. 53). Na apresentação dos produtos, o **Product Owner**, assessorado pela equipe técnica, verifica se todos os produtos previstos na solicitação de sustentação foram entregues (peça 20, p. 62). Posteriormente, os produtos entregues são avaliados pelo TST com vistas à emissão do termo de recebimento definitivo da Ordem de serviço (peça 20, p. 30).*

*217. As ordens de serviço das **sprints** relativas à sustentação programada são remuneradas conforme o esforço, dado em Horas de Serviço Técnico (HST). O esforço (E) é obtido a partir do produto do tamanho em pontos de função da **sprint** (T) pela produtividade estabelecida pelo TST (10 HST/ponto de função) para a contratada (E = T x P) (peça 20, p. 100).*

218. Além da prestação dos serviços de desenvolvimento propriamente dito, a contratada também é demandada em outras atividades, como recebimento e planejamento da ordem de serviço.

219. No recebimento, a contratada deve analisar a OS e verificar se as informações nela expressas são suficientes para a execução do serviço (peça 20, p. 27). No planejamento, a contratada deve estimar o esforço, prazo e custo necessários para a conclusão da OS (peça 20, p. 99). O esforço despendido pela contratada em ambos os serviços não é objeto de remuneração (peça 20, p. 32).

Banco Central do Brasil (Bacen)

220. No Contrato 50.412/2012 do Bacen, os serviços são demandados por Ordens de serviço cujo fluxo é apresentado em figura constante na página 27 do edital do pregão eletrônico Demap 7/2012 (peça 21). Segundo o fluxo estabelecido, o Banco cria e especifica o problema a ser resolvido na ordem de serviço e a envia para a contratada, a quem cabe a elaboração de proposta de solução. Estando de acordo com a proposta da contratada, o Bacen autoriza a execução da OS.

221. As ordens de serviço emitidas para o desenvolvimento de sistemas identificam qual o Processo ou metodologia deve ser usado pela contratada (peça 20, p. 29). Conforme demonstram as OS já emitidas no âmbito do Contrato 50.412/2012, o processo utilizado é o Processo de Desenvolvimento Ágil do Bacen (PDS-Ágil; peça 27, p. 55-469).

222. A contratada deverá entregar os produtos resultantes de uma ordem de serviço somente após a execução completa de todos os serviços nela requeridos. Contudo, a critério do Bacen, poderão ser acordadas entregas parciais de serviços para uma OS de projeto de construção. Uma entrega parcial em serviços de desenvolvimento e manutenção constitui-se de uma release de código, juntamente com a documentação associada, que implementa um conjunto de funcionalidades quantificáveis e que tenham significado para o solicitante do serviço (peça 21, p. 40).

223. A remuneração da contratada observa o estabelecido no documento Distribuição de Esforço por Disciplina (peça 27, p. 52-55). Quanto ao PDS-Ágil, ao elaborar o levantamento de requisitos e após tê-lo aprovado pelo Bacen, a contratada recebe 14,375% do valor correspondente

aos pontos de função estimados para a respectiva ordem de serviço. A fase de levantamento de requisitos compreende a elaboração do modelo de negócio, documento de visão, protótipo de interface, especificação de teste de aceitação e glossário (peça 27, p. 54).

224. Após a fase de implementação – a qual abrange a elaboração de documento de arquitetura, modelo de dados, classes de testes de unidade, código fonte, instrumentação de testes, script de teste, relatório de teste, ajuda do sistema, roteiro de operação e manual de usuário – a contratada faz jus aos 85,625% restantes, a depender do aceite definitivo dos produtos gerados (peça 27, p. 54).

225. A fase de validação da ordem de serviço divide-se em provisória e definitiva. A OS é provisoriamente validada quando for confirmada a entrega dos serviços prestados. Caso seja rejeitada, a OS é devolvida à contratada para os ajustes necessários, repetindo-se esse procedimento até ocorrer a validação provisória (peça 21, p. 42).

226. Uma vez validada provisoriamente, a ordem de serviço passa pela validação definitiva, a qual compreende a análise detalhada dos produtos e dos resultados gerados para os serviços requeridos na OS e a verificação do atendimento aos critérios de aceitação nela estabelecidos e segundo os padrões e processos de trabalho do Bacen (peça 21, p. 42).

227. Na validação definitiva, a ordem de serviço pode ser validada, validada com restrições ou rejeitada (peça 21, p. 42). É validada com restrições quando for identificada alguma ocorrência em um ou mais produtos ou resultados, mas que permitam a validação da OS pelo Bacen. Ordens de serviços validadas com restrições são devolvidas à contratada para ajustes, concedendo-se prazo de até 50% do originalmente previsto para sua execução. Por sua vez, a ordem de serviço rejeitada é devolvida à contratada para correção, havendo tantas devoluções e entregas quantas necessárias até a validação definitiva do serviço. A seu critério, o Bacen pode prorrogar o prazo de conclusão da ordem de serviço rejeitada, uma única vez, em até 50% do prazo originalmente previsto para sua execução (peça 21, p. 42-43).

<u>Instituto do Patrimônio Histórico e Artístico Nacional (Iphan)</u>

228. O Contrato 28/2011 previu o desenvolvimento do Sistema Integrado de Conhecimento e Gestão (SICG) de forma iterativa e incremental por meio de ciclos, com duração máxima de três semanas cada um, executados mediante ordens de serviço dimensionadas por meio de pontos de função brutos (peça 22, p. 27 e 29).

229. O primeiro ciclo estabelecido no termo de referência consiste no ciclo de planejamento, o qual é sucedido do ciclo de transição para os ciclos de desenvolvimento, ciclos de desenvolvimento, ciclo de transição para o ciclo de implantação, ciclo de implantação, ciclo de transição para o ciclo de encerramento e ciclo de encerramento (peça 22, p. 30).

230. A execução de cada ciclo é precedida por uma reunião de planejamento, na qual a contratada apresenta ao Iphan uma proposta de Ordem de serviço contendo o planejamento proposto para o ciclo. Caso aprovada pelo Iphan, a OS é devidamente formalizada e entregue à contratada para dar início às atividades previstas no ciclo (peça 22, p. 34).

231. Cada ciclo é finalizado pela reunião de encerramento, oportunidade em que os produtos resultantes da OS são apresentados (peça 22, p. 34). O prazo para a aceitação definitiva dos produtos entregues pelo Iphan, dada pela emissão de pareceres técnico e negocial favoráveis, consiste no prazo de execução do próximo ciclo (peça 22, p. 35).

232. A remuneração da contratada para os ciclos de desenvolvimento baseia-se na quantidade de pontos de função executados no ciclo (peça 22, p. 35). O termo de referência do Pregão Eletrônico 12/2011 previu que 75% do valor do contrato seriam destinados aos ciclos de desenvolvimento, ou seja, na realidade a contratada recebe apenas 75% do valor dos pontos de função brutos da contagem final de cada ciclo de desenvolvimento (peça 22, p. 37).

233. Por seu turno, aos ciclos de projeto (planejamento, implantação e encerramento), não mensuráveis por pontos de função, foram destinados 5%, 10% e 10% da quantidade estimada de dois mil pontos de função brutos. O valor pago referente ao ciclo de planejamento pode sofrer revisão,

uma vez que a quantidade final de pontos de função do SICG pode não corresponder aos dois mil pontos inicialmente estimados (peça 22, p. 37).

*234. Acerca da futura contratação de fábrica de **software** para o desenvolvimento e manutenção de sistemas informatizados do Iphan, o modelo de remuneração da contratada previsto na minuta do termo de referência assemelha-se ao do Contrato 28/2011.*

235. No novo contrato, as demandas serão solicitadas por meio de Ordens de Serviço, cuja remuneração será calculada considerando a quantidade de pontos de função a serem produzidos e a fase do desenvolvimento. Na fase de planejamento, caberá 5% dos pontos de função estimados, enquanto nas fases de desenvolvimento e encerramento caberá à contratada receber 80% e 15%, respectivamente, dos pontos de função efetivamente produzidos (peça 27, p. 496).

Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira (Inep)

236. No Contrato 33/2012, atualmente vigente, o fluxo do processo de desenvolvimento ou manutenção programada de sistemas é detalhado na Metodologia de Gestão e Desenvolvimento de Sistemas (MGDS; peça 24, p. 513-551). Segundo a MGDS, a definição da visão do novo sistema a ser desenvolvido ou de sistema legado a sofrer manutenção dá-se na etapa de iniciação, fase que pode ser executada com ou sem auxílio da contratada.

*237. A iniciação constitui-se no planejamento propriamente dito. Nessa fase, além da elaboração da visão do produto e do estabelecimento da definição de 'pronto', também são definidos os requisitos que constarão do **backlog** do produto (peça 24, p. 513). Caso haja necessidade de participação da contratada na iniciação, uma Ordem de serviço específica é aberta para a realização de atividades de levantamento preliminar de escopo junto ao cliente ou de detalhamento da demanda. Nesse tipo de ocorrência, a contratada tem remuneração correspondente a 10% da quantidade de pontos de função estimada para a respectiva OS (peça 24, p. 69).*

238. A fase de iniciação é encerrada quando a demanda planejada, seja desenvolvimento de novo sistema, seja manutenção de sistema legado, é repassada para a contratada com a finalidade de alinhar o entendimento do negócio entre o Inep e a contratada (peça 24, p. 519).

*239. Na sequência, inicia-se a fase de execução, que é desenrolada segundo os preceitos do **framework Scrum**, com reunião de planejamento da **sprint**, na qual é emitida OS para a sua execução pela contratada, e reunião de revisão, na qual a contratada entrega os produtos da **sprint**, além da reunião de retrospectiva (peça 24, p. 521-526). Para a execução da **sprint**, a contratada é remunerada em 100% do total obtido na medição final dos pontos de função da iteração entregue e aceita (peça 24, p. 69).*

*240. Ao final da fase de execução também pode se dar a implantação, no ambiente de produção, do produto gerado na **sprint**, quando aceito e homologado pelo Inep e solicitado pelo **Product owner** (peça 24, p. 526).*

*241. A última fase do processo de desenvolvimento ou manutenção de sistemas, de acordo com a MGDS, consiste no encerramento, que tem como objetivo encerrar as ordens de serviço de iniciação e da **sprint** (peça 24, p. 537-538). No encerramento, ocorre a aceitação definitiva das Ordens de Serviço, habilitando-as para faturamento pela contratada.*

*242. Antes da execução da fase de encerramento, a MGDS prevê a homologação dos artefatos produzidos pela contratada em atendimento às Ordens de serviço de iniciação e da **sprint** em dois momentos. No primeiro, a avaliação da qualidade dos artefatos é realizada pela área de TI quanto à conformidade com os padrões estabelecidos pelo Inep (peça 24, p. 528-530). No segundo, a avaliação é feita sob o ponto de vista do negócio pelo cliente demandante (peça 24, p. 535-536).*

Supremo Tribunal Federal (STF)

243. No Contrato 24/2013, as demandas são encaminhadas à empresa contratada por meio de ordens de serviço, as quais são formalizadas na reunião de abertura da OS (peça 25, p. 34).

*244. Após a reunião de abertura da OS, inicia-se a etapa de preparação da **sprint**, que serve, basicamente, para garantir o entendimento, por parte da contratada, do serviço solicitado e facilitar o planejamento da execução. Nessa etapa, é apresentado à contratada um subconjunto de histórias de*

*usuário, que consiste em parte do **product backlog**, para que sejam selecionadas aquelas que comporão o **backlog** da **sprint** (peça 25, p. 34).*

*245. Após a preparação, inicia-se a etapa de execução, a qual consiste na **sprint** propriamente dita para a construção do produto. O prazo da **sprint** é fixado em quinze dias, podendo ser alterado ao longo da vigência contratual (peça 25, p. 31). A execução, por sua vez, inicia-se com a reunião de planejamento da **sprint** e encerra-se com a reunião de demonstração (peça 25, p. 30).*

*246. Na reunião de planejamento, é criado o **backlog** da **sprint**, as histórias de usuário são detalhadas em tarefas a executar, a contratada assegura o entendimento técnico e negocial da demanda, o tamanho e critérios de aceitação são definidos e a definição de 'pronto' é estabelecida (peça 25, p. 35).*

*247. Na reunião de demonstração, os produtos construídos com base nas histórias de usuário selecionadas para a **sprint** são apresentados em ambiente de homologação. Nessa reunião, a contratada também deve entregar a documentação prevista no instrumento convocatório, a exemplo de documento de arquitetura, modelo de dados e plano de implantação. Nessa reunião, conforme interesse do STF, a contratada deve detalhar e repassar todo o conhecimento técnico utilizado na construção dos produtos (peça 25, p. 36 e 44).*

248. Sucede a etapa de execução a avaliação, período em que o produto é verificado funcional e tecnicamente (peça 25, p. 30). Não havendo inconformidades e ocorrências, o produto é homologado e a ordem de serviço originária é encerrada (peça 25, p. 38).

*249. O pagamento das ordens de serviço é calculado com base no tamanho funcional do produto homologado em pontos de função brutos e na efetiva execução do serviço. Ele é feito em duas etapas: a primeira, correspondente a 90%, é realizada após o recebimento definitivo da OS referente à última **sprint** de cada produto; a segunda, correspondente a 10%, é realizada somente ao final do período de garantia, estipulado em 180 dias (peça 25, p. 21 e 39).*

5.4. Mudanças de requisitos e escopo

*250. A filosofia dos métodos ágeis para desenvolvimento de **software** foi concebida para absorver mudanças, conforme expresso no manifesto ágil e em seus princípios, especificamente no valor 'responder a mudanças, mais que seguir um plano' e no princípio 'aceitar mudanças de requisitos, mesmo no fim do desenvolvimento'. Tal mudança pode acontecer inclusive durante as iterações, ciclos ou **sprints**. No **Scrum**, durante a **sprint**, não são feitas mudanças que possam afetar seu objetivo, porém o escopo pode ser esclarecido e renegociado entre o **Product Owner** e a equipe de desenvolvimento à medida que for melhor entendido.*

251. Entretanto, foi constatado que a maioria das instituições contratantes não permite mudanças durante a execução das iterações. Exemplo dessa prática pode ser visto na página 32 do edital do Pregão Eletrônico 12/2011 do Iphan (peça 22), na qual consta que 'uma vez que o ciclo tenha sido aprovado pela CONTRATANTE e a Ordem de serviço tenha sido emitida, nenhuma das partes poderá alterar o escopo do ciclo até que ele finalize'.

*252. Quando identificadas, as mudanças são colocadas no **backlog** do produto para que, em momento oportuno, conforme a priorização da área de negócios, sejam introduzidas em iterações seguintes.*

253. Por outro lado, no Bacen, a possibilidade de alteração do escopo de uma Ordem de serviço está explicitamente prevista no edital do Pregão Eletrônico Demap 7/2012 por meio do documento Solicitação de Mudança, no qual a mudança de escopo é solicitada, com respectiva análise de impacto no tamanho, prazo e custo (peça 21, p. 21).

5.5. Níveis de serviço

*254. Da análise dos editais das contratações, observa-se que quase todas as instituições visitadas instituíram vários níveis de serviço a serem atendidos pela contratada. Em suma, o não atendimento a critérios mínimos aceitáveis implica a redução do valor a ser pago à contratada. Dos níveis de serviço analisados, verificou-se a preocupação quase comum das contratantes com o prazo para o cumprimento das iterações, ciclos ou **sprints**, e com a qualidade dos produtos entregues.*

Tribunal Superior do Trabalho (TST)

255. No TST, por exemplo, foi instituído o Índice de Atraso na Entrega de OS, o qual tem por objetivo garantir a entrega da OS no prazo estimado para a sua conclusão (peça 20, p. 37). Embora a meta estabelecida para esse indicador seja de 0%, o TST estipulou que atrasos inferiores a 10% do prazo previsto para a entrega da OS encontram-se em um limite aceitável. Isto é, no contrato do TST há tolerância para atrasos na entrega dos produtos demandados, afastando-se dos preceitos do ***framework Scrum****, no qual uma* ***sprint*** *deve ser executada no prazo estabelecido.*

256. Quanto à aferição da qualidade, o TST criou o indicador Desconformidade na OS (DOS) para assegurar a conformidade das Ordens de serviço aos requisitos de qualidade, determinados nos modelos a serem utilizados para a elaboração dos artefatos produzidos pela contratada, bem como nas listas de verificação para a avaliação desses artefatos (peça 20, p. 63, 65-95). A meta do indicador DOS é 0%, significando que caso alguma desconformidade seja detectada, o produto, e, consequentemente, a Ordem de serviço, é rejeitada.

Banco Central do Brasil (Bacen)

257. Por sua vez, o Bacen instituiu indicador referente ao Atraso na Entrega da OS de projeto de construção, tolerando, assim como o TST, pequenos atrasos, uma vez que esse indicador deve ser menor ou igual a 0,1 (peça 21, p. 63). Para aferir a qualidade dos artefatos entregues pela contratada em projetos de construção, o Bacen definiu o Índice de Defeitos por Pontos de Função da OS e o Índice de Defeitos Impeditivos por Pontos de Função da OS. Para o primeiro indicador, há tolerância para a aceitação da OS, enquanto para o segundo a tolerância é zero (peça 21, p. 63-64).

Instituto do Patrimônio Histórico e Artístico Nacional (Iphan)

258. De forma diferente das outras instituições visitadas, o Iphan, no Contrato 28/2011, institui apenas dois indicadores de nível de serviço, sendo que somente um deles refere-se ao serviço de desenvolvimento, executado nos ciclos de desenvolvimento, o qual é denominado de Índice de Pontos Executados (IPE). O IPE destina-se a avaliar a qualidade dos produtos entregues, medindo a quantidade de pontos de função brutos executados e aceitos da ordem de serviço. Caso o IPE seja inferior ou igual a 80%, multas podem ser aplicadas à contratada (peça 22, p. 45-46).

259. A instituição do IPE no instrumento convocatório do Iphan não se confunde com a aceitação de Ordens de serviço contendo produtos em desconformidade com os padrões de qualidade estabelecidos. Caso algum produto construído no ciclo não seja aprovado, este poderá ser considerado perdido, devendo um novo ciclo ser iniciado para adequação desse produto (peça 22, p. 33).

260. Acerca do atraso na entrega dos produtos demandados nas ordens de serviço, ou seja, atraso nos ciclos, o Iphan não prevê tal situação no termo de referência do Pregão Eletrônico 12/2011. No Contrato 28/2011, ao final de cada ciclo, a contratada deve entregar os produtos que porventura tenha produzido, os quais serão avaliados pelo IPE. Caso não tenha produtos a entregar, assim como os produtos não aprovados, eles deverão ser objeto de outro ciclo.

261. O edital em elaboração do Iphan para a contratação de fábrica de ***software*** *para desenvolvimento e manutenção de sistemas com métodos ágeis também mantém o mesmo indicador (IPE) para a avaliação dos produtos entregues (peça 27, p. 496-497).*

Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira (Inep)

262. No Inep, no âmbito do Contrato 33/2012, os critérios de avaliação da qualidade dos produtos entregues possuem indicadores distintos para a avaliação a cargo da área de TI e a da área de negócios. A área de TI avalia a conformidade da qualidade dos produtos sob o aspecto técnico, homologando-os tecnicamente. A área de negócios avalia a conformidade dos produtos sob o aspecto negocial, homologando-os negocialmente. Segundo o instrumento convocatório, havendo não conformidades nessas etapas, são concedidos prazos (48 horas na homologação técnica e 24 na homologação negocial) consecutivos para a contratada adequar os produtos à qualidade exigida, avaliando-se negativamente a contratada a cada novo prazo dado (peça 24, p. 55-56, Tabela 1, itens 1, 2, 4 e 5).

263. Acerca do cumprimento do prazo para a execução das Ordens de Serviço, segundo o termo de referência, há tolerância quando ocorre o descumprimento do cronograma estabelecido para a conclusão dos serviços. Entretanto, a contratada é avaliada de forma negativa quando atrasa a entrega dos produtos especificados na Ordem de serviço, concedendo-se sucessivos prazos de 24 horas para finalizá-los, nos quais a penalização da contratada é agravada (peça 24, p. 56, Tabela 1, itens 6 e 7).

Supremo Tribunal Federal (STF)

264. O STF aplica dois tipos de redutores às Ordens de serviço pagas à empresa contratada. O primeiro refere-se aos níveis de serviço propriamente ditos, enquanto o segundo relaciona-se aos critérios gerais de avaliação (peça 25, p. 52-53). Entre os níveis de serviço, encontra-se o Índice de Desconformidade do Produto, o qual afere a qualidade dos produtos entregues pela contratada. Por sua vez, nos critérios gerais de avaliação consta a punição a ser aplicada à contratada quando ocorrer atraso não justificado nas datas definidas nas Ordens de Serviço.

*265. Ainda chama a atenção no Contrato 33/2012, do Inep, e no Contrato 24/2013, do STF, a existência de indicadores específicos de rotatividade da equipe da contratada (peça 24, p. 56-57, item 10 e peça 25, p. 47 e 52, item 1). Uma vez que o desenvolvimento de **softwares** com a utilização de métodos ágeis tem foco voltado para os indivíduos, em vez de processos, e na entrega de resultado de forma adiantada, a substituição de membros da equipe da contratada, como o **Scrum Master** ou membros da equipe de desenvolvimento, pode efetivamente diminuir a produtividade do fornecedor.*

5.6. Compatibilização dos valores ágeis nas contratações examinadas

266. O exame dos contratos das instituições visitadas demonstra o ânimo de impor a essência das metodologias ágeis, postuladas pelos valores do Manifesto Ágil, às contratações realizadas, compatibilizando-a com as normas vigentes.

*267. Nesse sentido, constatou-se a preocupação de todas as instituições com relação à entrega de artefatos de documentação associados ao **software** produzido a cada iteração, facilitando, por exemplo, futuras manutenções por terceiros alheios ao processo de desenvolvimento. Também foi constatado que a relação contratual prevalece sobre a possível colaboração entre as partes, em harmonia com o princípio da vinculação ao instrumento convocatório, e as mudanças propostas mostraram-se restritas a novas iterações, mitigando o risco de desembolsos não programados.*

*268. Entretanto, quanto à pessoalidade – forte aspecto das metodologias ágeis, fundamentado na maior valorização dos indivíduos e interação entre eles em detrimento de processos e ferramentas, e materializado na necessidade da constância da composição da equipe de desenvolvimento do time **Scrum** (item 90) – foi constatada, pela existência de níveis de serviço vinculados à rotatividade da equipe de desenvolvimento da contratada em contratos de duas instituições, a preocupação com essa característica do método.*

6. Riscos na contratação de desenvolvimento de software com métodos ágeis pelas instituições da Administração Pública Federal

*269. Com o conhecimento adquirido sobre métodos ágeis, seja decorrente da análise teórica ou das visitas realizadas pela equipe de fiscalização a algumas instituições públicas federais, seja decorrente do exame dos contratos dessas instituições, é possível vislumbrar alguns riscos nas contratações públicas para desenvolvimento de **software** por meio de métodos ágeis.*

*270. Nesse sentido, a seguir são apresentados alguns riscos inerentes ao novo paradigma de desenvolvimento de **software** que está sendo difundido nas contratações da Administração Pública, suas possíveis causas e consequências, bem como contratações em que foram materializados, quando for o caso. Para fins didáticos, os riscos elencados estão reunidos em três grupos distintos: processos, produtos e pessoas.*

*271. Cumpre frisar que não se trata de enumeração exaustiva de riscos, e sim de um subconjunto identificado com o conhecimento adquirido. Também é importante observar que alguns dos riscos expostos não são inerentes somente ao uso de métodos ágeis, podendo ocorrer também com metodologias tradicionais de desenvolvimento de **software**.*

Riscos relativos a processos

*272. Risco 1: contratação de desenvolvimento de **software** com adaptação de metodologia ágil que desvirtue sua essência.*

272.1. Uma vez que a utilização estrita da doutrina ágil pode ser conflitante com algum dispositivo constante dos diversos normativos e jurisprudência vigentes ou não atender à totalidade dos interesses da instituição pública contratante, esta pode incorrer em adaptações de uma metodologia ágil já consolidada no mercado com o intuito de moldá-la à sua realidade. Essa prática pode ter como consequências, entre outras:

272.1.1. diminuição da competitividade da licitação, pois a utilização de metodologias adaptadas pode não despertar o interesse de fornecedores que já adotam uma metodologia específica, preferindo não realizar contratos que apresentem risco para o seu negócio;

272.1.2. conflitos com a empresa contratada, uma vez que podem surgir interpretações equivocadas pelas partes quanto ao emprego dos processos, atividades e papeis estabelecidos no instrumento convocatório, não havendo organização que suporte a metodologia adaptada, a qual possa ser utilizada para dirimir eventuais dúvidas e, por consequência, eliminar os conflitos;

272.1.3. conflitos entre membros da instituição contratante devido a falhas na atribuição dos papeis a serem desempenhados;

*272.1.4. entrega de produtos de baixa qualidade, caso a adaptação da metodologia suprima elementos do processo de elaboração do **software** servíveis à construção e à aferição da qualidade do produto;*

272.1.5. diminuição da eficiência/produtividade da equipe de desenvolvimento da empresa contratada, caso a adaptação feita na metodologia anule ou atenue práticas que visem à eficiência;

*272.2. Algumas das consequências vislumbradas no risco em destaque podem vir a ocorrer no Contrato 146/2012 do TST. O **framework** adotado nessa instituição é o **Scrum**. Entretanto, segundo o subitem 3.4 (peça 20, p. 50), vários sistemas podem ser agrupados em uma mesma **sprint**:*

*'3.4 O Processo de Sustentação Programada será executado na forma de **Sprints**, como definido no método de desenvolvimento ágil **Scrum**, com a principal diferença de permitir o agrupamento de manutenções de vários sistemas diferentes em um mesmo **Sprint**.' (grifo nosso)*

*272.3. uma vez que o mesmo time de desenvolvimento **Scrum** deve estimar e realizar a contagem dos pontos de função relativos a novas funcionalidades de mais de um sistema, gerenciar a execução de atividades diversas pulverizadas em vários sistemas, interagir com **POs** distintos durante a execução de uma única **sprint**, sua eficiência pode sofrer prejuízos.*

*272.4. O TST também inovou ao especificar, no termo de referência, que o papel do **Product Owner** pode ser desempenhado por mais de uma pessoa (peça 20, p. 56):*

'9.1 O Dono do Produto pode ser um profissional, ou um grupo de profissionais, da CDS e/ou das áreas de negócio do TST.'

*272.5. Nesse caso, as posições dos donos do produto podem ser divergentes, gerando, por exemplo, problemas na gerência do **backlog** do produto e na avaliação das funcionalidades entregues.*

*272.6. Por seu turno, a Metodologia Iphan de Gestão de Demandas de Desenvolvimento Ágil de **Softwares** (Midas), a qual será adotada na futura contratação de fábrica de **software** que se avizinha, define a existência de dois **Scrum Masters** no modelo de gerência do desenvolvimento de **software**, sendo um da contratada e o outro do próprio Iphan. No **framework Scrum**, é esperado que tal papel seja desempenhado pela contratada. Tal fato pode gerar problemas de competência com a empresa contratada.*

273. Risco 2: alteração da metodologia ágil adotada no instrumento convocatório no decorrer da execução contratual.

273.1. A alteração da metodologia ágil adotada no instrumento convocatório durante a execução contratual pode ocorrer devido à pouca experiência da instituição pública contratante na utilização de métodos ágeis. Essa prática pode ter como consequências conflitos de ordem financeira com a empresa contratada, uma vez que o seu preço, apresentado à época da licitação, pode vir a não

cobrir novos custos decorrentes de alterações introduzidas pela contratante, trazendo distorções à manutenção do equilíbrio econômico-financeiro do contrato. Além disso, afronta o princípio da vinculação ao instrumento convocatório instituído no art. 3º da Lei 8.666/1993.

273.2. Uma variação do risco em epígrafe é a inclusão de cláusulas e condições no instrumento convocatório e no contrato dele decorrente, as quais prevêem que a metodologia especificada pode sofrer alterações durante a vigência contratual. Nesse caso, além de trazer distorções à manutenção do equilíbrio econômico-financeiro do contrato, tal prática pode configurar afronta ao art. 14 da lei de licitações e contratos, que veda a contratação sem a adequada caracterização de seu objeto.

273.3. No Bacen, algumas das cláusulas do instrumento convocatório que originou o Contrato 50.412/2012 apresentam essa característica, como exemplificado a seguir.

'8.1.4 A critério do Bacen os padrões, processos de trabalho e artefatos poderão sofrer alterações. A Contratada deverá se adaptar às mudanças no prazo máximo de 30 dias corridos contados da comunicação pelo Bacen.' (peça 21, p. 25)

*'8.1.5 A critério do Bacen, durante a execução contratual, poderão ser acrescentadas ao conjunto de processos de desenvolvimento, manutenção e documentação vigentes, outras metodologias, práticas, artefatos e tecnologias (**frameworks**, ambiente operacional e de desenvolvimento, arquitetura dentre outros) que sejam aderentes às formas de mensuração, de pagamento e de serviços previstas neste Edital.' (peça 21, p. 24-25)*

'17.2.1.2 O Bacen poderá estipular, na própria Ordem de serviço, em função de suas características específicas, outros critérios de validação dos serviços, adicionais ou substitutivos aos descritos nos padrões e processos de trabalho do Bacen.' (peça 21, p. 42)

273.4. As alterações previstas nesses casos, quanto ao desenvolvimento de sistemas no Bacen, afetam basicamente o Processo de Desenvolvimento Ágil do Bacen (PDS-Ágil).

274. Risco 3: ausência de definição dos artefatos ou alteração dos artefatos exigidos da contratada no instrumento convocatório durante a execução contratual.

274.1. O risco em evidência, assim como o anteriormente citado, pode decorrer da pouca experiência da instituição pública contratante na utilização de métodos ágeis.

*274.2. A ausência da definição dos artefatos exigidos da contratada no instrumento convocatório pode elevar os custos da contratação, uma vez que a empresa contratada, por não conhecer a totalidade dos produtos demandados para a construção do **software** à época da licitação, pode apresentar proposta de preços majorada em virtude da insegurança inerente ao desconhecimento.*

274.3. além de ser potencialmente lesiva à economicidade da contratação, a materialização do risco em epígrafe pode configurar afronta ao art. 14 da lei de licitações e contratos, que veda a contratação sem a adequada caracterização de seu objeto.

274.4. Por sua vez, a alteração dos artefatos exigidos da contratada no instrumento convocatório no decorrer da execução contratual afronta o princípio da vinculação ao instrumento convocatório instituído no art. 3º da Lei 8.666/1993 e pode gerar conflitos com a empresa contratada por vir a introduzir novos custos não vislumbrados no edital, ferindo a manutenção do equilíbrio econômico-financeiro do contrato. Em acréscimo, também fere o art. 14 da lei citada.

274.5. No Bacen, os subitem 8.1.4 e 8.1.5 do termo de referência do Pregão Eletrônico Demap 7/2012, transcritos a seguir, apresentam essa característica:

'8.1.4 A critério do Bacen os padrões, processos de trabalho e artefatos poderão sofrer alterações. A Contratada deverá se adaptar às mudanças no prazo máximo de 30 dias corridos contados da comunicação pelo Bacen.' (peça 21, p. 25)

*'8.1.5 A critério do Bacen, durante a execução contratual, poderão ser acrescentadas ao conjunto de processos de desenvolvimento, manutenção e documentação vigentes, outras metodologias, práticas, artefatos e tecnologias (**frameworks**, ambiente operacional e de*

desenvolvimento, arquitetura dentre outros) que sejam aderentes às formas de mensuração, de pagamento e de serviços previstas neste Edital.' (grifo nosso) (peça 21, p. 24-25)

275. Risco 4: exigência de artefatos desnecessários ou que se tornam obsoletos rapidamente.

*275.1. A exigência de artefatos desnecessários pode ser oriunda da inexperiência da instituição contratante, principalmente na transição do modelo de contratação para desenvolvimento de **software** com metodologias tradicionais para métodos ágeis.*

*275.2. Alguns artefatos exigidos no instrumento convocatório podem, no novo paradigma, não trazer utilidade à contratante ou tornarem-se obsoletos rapidamente, de uma **sprint** para outra, como manual do usuário. Além disso, acrescentam custos à empresa contratada, os quais possivelmente foram repassados em sua proposta de preços, onerando de forma desarrazoada o ajuste contratual. Dessa forma, a materialização do risco em tela afronta o princípio constitucional da economicidade.*

*276. Risco 5: utilização de contrato para desenvolvimento de **software** por metodologias tradicionais para desenvolvimento por métodos ágeis.*

*276.1. A utilização de contrato inicialmente concebido para ser executado segundo modelo de desenvolvimento de **software** tradicional para o desenvolvimento de **software** por meio de métodos ágeis, em primeira análise, conflita com o princípio da vinculação ao instrumento convocatório, estabelecido no art. 3º da Lei 8.666/1993. Nesse caso, trata-se de alteração no objeto do serviço de desenvolvimento de **software**, haja vista que a utilização de métodos ágeis pode alterar, em forma ou em essência, os produtos inicialmente descritos no contrato.*

263.2. Potencialmente, a materialização do risco em tela pode gerar atritos entre a instituição contratante e a fornecedora devido à mudança do paradigma utilizado na execução contratual, não previsto inicialmente no instrumento convocatório. Em adição, é possível que vários elementos constantes no ajuste contratual não se adequem à utilização de métodos ágeis, como níveis de serviço e artefatos exigidos da contratada no decorrer da execução contratual.

Riscos relativos a pessoas

*277. Risco 6: falta de comprometimento ou colaboração insatisfatória do responsável indicado pela área de negócios (**Product Owner**) no desenvolvimento do **software**.*

*277.1. O uso de métodos ágeis exige grande comprometimento do responsável indicado pela área de negócios da instituição pública, conhecido como **Product Owner** no **framework Scrum**. A ele é atribuído o gerenciamento do produto de forma a assegurar o valor do trabalho executado pela equipe de desenvolvimento da contratada.*

277.2. Para que o processo de construção do produto possa fluir adequadamente, o responsável indicado pela área de negócios deve desempenhar diversas atividades, como definir a visão do produto, gerenciar suas funcionalidades, priorizá-las e aprovar ou rejeitar os artefatos entregues a cada iteração. Além disso, deve ter disponibilidade para atender, sempre que necessário, a equipe de desenvolvimento da contratada para que essa possa, por exemplo, elidir dúvidas emergentes.

*277.3. A falta de comprometimento ou a colaboração insatisfatória do responsável no processo de construção do **software**, abdicando do exercício das atividades a ele atribuídas, pode ter como consequências a geração de produtos de baixa qualidade que não atendam às reais necessidades dos clientes, atrasos no desenvolvimento e até mesmo, em casos extremados, o cancelamento do projeto.*

*278. Risco 7: falta do conhecimento necessário do indicado pela área de negócios (**Product Owner**) para o desenvolvimento do **software**.*

*278.1. O servidor indicado pela área de negócios responsável pela construção do **software** para desempenhar o papel de **Product Owner** ou similar pode não deter os conhecimentos necessários dos processos de trabalho que serão apoiados pela solução de TI a ser desenvolvida, podendo acarretar definições e priorizações de funcionalidades equivocadas, em detrimento das funcionalidades essenciais.*

*278.2. Em acréscimo, outras consequências potenciais da falta de conhecimento do **Product Owner** indicado são esforços e custos posteriores para ajustar a solução mal concebida ou, até mesmo, o seu descarte, afrontando os princípios constitucionais da economicidade e da eficiência.*

*278.3. No caso da materialização desse risco, pode caber responsabilização do **Product Owner** e do responsável pela sua indicação quanto aos recursos despendidos sem que o interesse público tenha sido atendido.*

278.4. No Iphan, em entrevista com o responsável pelo Contrato 28/2011 e a equipe de fiscalização, foi revelado que o risco ora tratado materializou-se. Na ocasião, o fato foi devidamente documentado nos autos do processo da contratação e comunicado à instância superior para o adequado tratamento.

*279. Risco 8: excessiva dependência da visão do indicado pela área de negócios (**Product Owner**).*

*279.1. A falta de interação do **Product Owner** com os demais usuários do **software** em construção e, até mesmo, o desconhecimento desses acerca do projeto, pode vir a criar excessiva dependência de sua visão na concepção do produto. Como efeito desse risco, o sistema de informação construído pode não atender às expectativas dos usuários e, por consequência, não atender à necessidade da contratação.*

*266.2. Outro aspecto relacionado à dependência excessiva do **Product Owner** refere-se a seus afastamentos da instituição em decorrência, por exemplo, de férias e licenças. Nesse caso, uma possível consequência é a suspensão da execução do projeto, acarretando atrasos na entrega do produto final. No Iphan, segundo relatos, quando o **Product Owner** afastou-se, em decorrência de férias, o andamento do projeto executado no âmbito do Contrato 28/2011 foi suspenso. Nessa ocasião, a empresa contratada aproveitou o período de afastamento do PO para introduzir melhorias técnicas no **software**.*

*280. Risco 9: equipe da empresa contratada não ter expertise em desenvolvimento de **software** com métodos ágeis.*

280.1. Em licitações públicas, para mitigar o risco da licitante vencedora do certame não possuir a capacidade técnica necessária para a execução do objeto, a Lei 8.666/1993 provê, em seu art. 30, mecanismos para que a futura contratada comprove estar tecnicamente apta para a prestação dos serviços. Isso se dá, notadamente, por meio da apresentação de atestados de capacidade técnica que demonstrem a execução de serviços pertinentes e compatíveis em características, quantidades e prazos com o objeto da licitação.

*280.2. Em relação ao risco em destaque, ainda que a licitante apresente os atestados exigidos no instrumento convocatório comprovando que já produziu sistemas utilizando métodos ágeis, e ainda que se façam diligências para comprovar sua veracidade, há a possibilidade de que os resultados demonstrados por esses documentos não sejam reproduzidos na nova contratação. Isso pode acontecer em virtude, por exemplo, de alocação, pela contratada, de equipe inexperiente, principalmente em contratações de fábrica de **software**.*

280.3. As consequências possíveis, quando da materialização do risco ora exposto, são atrasos constantes na entrega dos produtos e geração de produtos de baixa qualidade, resultando, em última análise, no não atendimento da necessidade da contratação.

*280.4. Na primeira experiência do Inep de contratação de desenvolvimento de **software** com métodos ágeis (Contrato 1/2011 – peça 23, p. 383), o risco em epígrafe materializou-se, conforme relatado por membros da área de TI à equipe de fiscalização. A equipe da empresa contratada alocada para a prestação do objeto do contrato não detinha o conhecimento necessário para o cumprimento dos serviços demandados, embora o instrumento convocatório exigisse dos perfis profissionais da contratada conhecimento em **Scrum**, entre outros (peça 23, p. 45-54).*

*281. Risco 10: dificuldade de comunicação entre a equipe de desenvolvimento da contratada com o indicado pela área de negócios (**Product Owner**).*

281.1. A maior valorização de indivíduos e a interação entre eles em detrimento de processos e ferramentas, expressa no Manifesto Ágil; o princípio no qual pessoas relacionadas a negócios e desenvolvedores devem trabalhar em conjunto e diariamente, durante todo o curso do projeto; e o princípio no qual é estabelecido que o método mais eficiente e eficaz de transmitir informações para, e por dentro de um time de desenvolvimento, é por meio de uma 'conversa cara a cara', deixam evidente que a comunicação entre as pessoas envolvidas no processo de desenvolvimento do ***software*** *tem importância fundamental.*

281.2. Conforme preceituado pelo valor Comunicação do ***eXtreme Programming (XP)****, diálogos presenciais são mais eficazes que videoconferências, que, por sua vez, são melhores que telefonemas, sendo esses mais expressivos que emails e assim sucessivamente (item 122.2). O diálogo presencial evita que problemas de má compreensão e ambiguidades comprometam negativamente o produto final.*

*281.3. Ainda que o indicado pela área de negócios (****Product Owner****) da instituição contratante tenha disponibilidade para atender o time de desenvolvimento da contratada, é necessária a existência de canais de comunicação eficazes entre ambos. Nesse sentido, a alocação da equipe da contratada nas instalações da contratante tende a facilitar o processo de comunicação, ao passo que a operacionalização dos serviços de desenvolvimento em uma localização remota tende a dificultá-lo.*

268.4. A dificuldade de comunicação entre os membros do time de desenvolvimento com o indicado pela área de negócios tem como potenciais consequências elaboração de produtos de baixa qualidade, atrasos na entrega dos produtos e, em última análise, traduz-se no não atendimento da necessidade da contratação.

Riscos relativos a produtos

282. Risco 11: alteração constante da lista de funcionalidades do produto.

282.1. Uma vez que a mudança de requisitos durante o projeto é bem aceita no desenvolvimento de ***software*** *com métodos ágeis, conforme se constata da leitura do Manifesto Ágil e de seus princípios, a lista de funcionalidades do produto pode ser constantemente alterada para incluir, ainda no desenvolvimento, novas características inicialmente não planejadas, previstas ou vislumbradas.*

282.2. A alteração constante e descontrolada da lista de funcionalidades do produto durante o contrato pode levar a instituição contratante a exceder prazos e custos de desenvolvimento preliminarmente estimados. Por consequência, a materialização do risco ora em destaque pode conduzir à execução de desembolsos excessivos, contrapondo-se ao princípio constitucional da economicidade e ao princípio do planejamento, bem como a atrasos na entrega do produto final ao cliente (área demandante), opondo-se ao princípio constitucional da eficiência.

283. Risco 12: iniciação de novo ciclo sem que os produtos construídos na etapa anterior tenham sido validados.

283.1. O processo de construção do ***software*** *por métodos ágeis comumente dá-se de forma contínua ao longo de ciclos, iterações ou* ***sprints****, nos quais um conjunto de funcionalidades é implementado. Algumas vezes, as funcionalidades a serem implementadas em um ciclo são dependentes do funcionamento de outras, implementadas em ciclos anteriores.*

283.2. Nesse tipo de situação, é possível que o ciclo no qual as funcionalidades previamente implementadas ainda não tenha sido validado, isto é, o produto gerado ainda não teve seu recebimento definitivo efetivado pela instituição contratante. Caso esse produto não seja aceito, o novo ciclo cujas funcionalidades são interdependentes e já iniciado fica prejudicado, ocasionando potenciais atrasos na construção do ***software****.*

284. Risco 13: falta de planejamento adequado do ***software*** *a ser construído.*

284.1. A doutrina ágil pode levar instituições públicas com equipes inexperientes ou sem nível de conhecimento técnico adequado ao entendimento equivocado de seu uso, relegando o adequado planejamento do produto a ser construído. O planejamento pode ser iniciado, de forma global, pela

*elaboração de documento da visão do produto e, de forma específica, pela preparação do **backlog** do produto (**grooming**), conforme estabelecido, por exemplo, no **framework Scrum** (item 110).*

*284.2. A área técnica da instituição contratante, por exemplo, pode entender que as fases de levantamento e de definição dos requisitos do **software** devem ocorrer, unicamente, durante as iterações de desenvolvimento. Tal entendimento pode ter como consequências possíveis a falta de consistência e de detalhamento da lista de funcionalidades a serem desenvolvidas (**backlog** do produto), bem como a necessidade de alterações do produto, comprometendo sua qualidade e elevando o custo do projeto.*

*285. Risco 14: pagamento pelas mesmas funcionalidades do **software** mais de uma vez, em virtude de funcionalidades impossíveis de serem implementadas em um único ciclo, ou em virtude da alteração de funcionalidades ao longo do desenvolvimento do **software**.*

*285.1. A construção do **software** utilizando métodos ágeis usualmente dá-se em ciclos, iterações ou **sprints**, os quais possuem prazo fixo para seu término (time-box). Nesses ciclos, funcionalidades do produto são selecionadas para serem implementadas. Podem existir funcionalidades cujo prazo da iteração não seja suficiente para sua implementação completa, sendo necessários outros ciclos para a sua conclusão. Nesses casos, os ciclos tratarão de desenvolver novas funcionalidades para aquelas parcialmente implementadas. Também é possível que, para a implementação de uma nova funcionalidade, uma outra já implementada necessite ser ajustada.*

*285.2. Essas situações, caso não previstas adequadamente no instrumento convocatório, podem fazer com que a instituição contratante efetue pagamentos pelas mesmas funcionalidades do **software** repetidas vezes, conflitando com o princípio constitucional da economicidade.*

*286. Risco 15: não disponibilização do **software** em ambiente de produção para a utilização e avaliação dos reais usuários.*

*286.1. Um dos objetivos dos métodos ágeis é a satisfação do cliente por meio da entrega adiantada e contínua de **software** funcional.*

*286.2. Uma das formas de avaliar a satisfação do cliente é disponibilizando o **software** no ambiente de produção, mesmo que para um reduzido grupo de usuários, em um projeto piloto. O objetivo dessa ação é permitir que os usuários validem e opinem acerca das funcionalidades do produto, construídas predominantemente segundo a visão do indicado pela área de negócios (**Product Owner**).*

*286.3. O uso do **software** pelos reais usuários possibilita a verificação de sua aderência às necessidades do negócio. Dessa forma, a demora na disponibilização do **software** para a validação junto aos usuários tem como consequência potencial a detecção tardia de erros de concepção, com consequente desperdício de recursos e esforço.*

*273.4. Quando a equipe de fiscalização visitou o Bacen, constatou-se que o Banco já tinha finalizado alguns projetos no âmbito do Contrato 50.412/2012, nos quais foram gerados **softwares** que ainda não tinham sido disponibilizados no ambiente de produção.*

287. Risco 16: forma de pagamento não baseada em resultados.

*287.1. A métrica popularmente adotada nas contratações para produção de **software** pelas instituições públicas é o ponto de função. O ponto de função mede o tamanho funcional do **software**, e não o esforço envolvido em sua concepção e construção.*

287.2. Esse fato é comumente criticado pelas empresas fornecedoras, uma vez que a utilização do ponto de função para remunerar os produtos por elas construídos pode não ser compatível com o esforço despendido e, por consequência, com os recursos financeiros por ela consumidos em sua produção.

*287.3. No âmbito de contratações privadas, é comum a utilização de outras formas de remuneração do fornecedor, como pagamento por iterações ou **sprints** e pagamento por linhas de código (**Source Lines of Code – SLOC**), que consiste em uma métrica utilizada para mensurar o tamanho de um **software**. Cumpre observar que algumas das formas adotadas na esfera privada não*

*consideram o resultado gerado pelo produto entregue. A remuneração da contratada por iterações ou **sprints**, por exemplo, constitui-se, na realidade, em pagamento por disponibilidade de mão de obra.*

287.4. De forma a utilizar especificações usuais praticadas no mercado, conforme preceitua o §2º do art. 3º do Decreto 3.555/2000, o qual regulamenta o pregão, a instituição contratante pode ver-se impelida a instituir em seu instrumento convocatório modelo de remuneração desvinculado da mensuração por resultados, em afronta à farta jurisprudência deste Tribunal de Contas, a qual se encontra consolidada na Súmula TCU 269:

'Nas contratações para a prestação de serviços de tecnologia da informação, a remuneração deve estar vinculada a resultados ou ao atendimento de níveis de serviço, admitindo-se o pagamento por hora trabalhada ou por posto de serviço somente quando as características do objeto não o permitirem, hipótese em que a excepcionalidade deve estar prévia e adequadamente justificada nos respectivos processos administrativos.' (grifo nosso)

*274.5. Uma possível consequência da materialização do risco do pagamento somente pela disponibilização de mão de obra reside no recebimento de produtos de **software** sem qualidade, resultando no não atendimento da necessidade da contratação.*

7. Conclusão

*288. O conhecimento adquirido neste levantamento permitiu entender a essência que orienta as metodologias ágeis de desenvolvimento de **software**, as quais voltam seu foco, primordialmente, para o atendimento das necessidades do cliente por meio da entrega contínua de **softwares** funcionais e de qualidade.*

*289. Por sua vez, as análises empreendidas no decorrer da execução desta fiscalização demonstram a viabilidade da adoção de metodologias ágeis em contratações destinadas ao desenvolvimento de **software** pela Administração Pública Federal (APF), assim como outras tantas metodologias que têm sido amplamente utilizadas ao longo dos últimos anos.*

290. Como em todo processo de contratação, há riscos que precisam ser considerados e mitigados. Contudo, no caso específico de adoção de métodos ágeis, tratados como novidade no mercado especializado nacional, sobretudo no âmbito da APF, a gestão de riscos inerentes às características do método merece atenção especial, no sentido de possibilitar que as instituições públicas possam fazer uso das práticas previstas sem incorrer em descumprimento dos normativos vigentes.

291. Por fim, cumpre registrar que o presente trabalho de fiscalização cumpriu o objetivo inicialmente vislumbrado, consistindo seu benefício em servir de instrumento que suporte eventuais fiscalizações que tratem desse tema a serem realizadas por esta unidade técnica especializada, não havendo benefícios pecuniários.

8. Proposta de encaminhamento

292. Diante do exposto, propõe-se o encaminhamento dos autos ao gabinete do Ministro José Múcio Monteiro com as propostas que seguem:

*292.1. levantar o sigilo deste processo em virtude de conter informações relevantes às instituições públicas quanto às contratações de serviços de desenvolvimento de **software** utilizando métodos ágeis, mantendo-se o sigilo da peça 27 destes autos, uma vez que contém documentos que não foram tornados públicos pelas respectivas instituições proprietárias;*

292.2. enviar, para conhecimento, cópia do acórdão que vier a ser proferido à Secretaria de Soluções de TI e à Secretaria de Controle Externo de Aquisições Logísticas, ambas vinculadas a este Tribunal de Contas;

292.3. arquivar os presentes autos, com fulcro no art. 169, inciso V, do RITCU."

É o relatório.