Quaderns de Psicologia | **2018**, Vol. 20, **No 2**, 115-127 | ISNN: 0211-3481

https://doi.org/10.5565/rev/qpsicologia.1433

Diálogos sobre a taylorização da atividade dos desenvolvedores das fábricas de software

Dialogues on the taylorization of the activity of developers from software factories

Dímitre Sampaio Moita
Cássio Adriano Braz de Aquino
Verônica Siqueira Araújo
Universidade Federal do Ceará

Resumo

Neste artigo debatemos a hipótese da taylorização do trabalho dos desenvolvedores das fábricas de software a partir do discurso destes profissionais. O uso de normas de qualidade na produção de software é debatido como possível traço de precariedade dentro deste contexto. Os argumentos de que o processo de software é rotinizado e os trabalhadores desqualificados são discutidos em contraponto com as falas dos próprios desenvolvedores. Consideramos a hipótese da taylorização imprecisa para descrever a experiência laboral dos entrevistados; visto que os pressupostos da organização taylorista não pautam a divisão da produção das fábricas de software. Logo, sendo mais apropriado refletir sobre o processo de racionalização da produção de software.

Palavras-chave: **Trabalho; Desenvolvedores de software; Fábricas de software; Taylorização**

Abstract

In this paper we discuss the hypothesis of taylorization of the work of the software factory developers from the discourse of these workers. The use of quality standards in software production is discussed as a possible trait of precariousness within this context. The arguments that the software process is routinized and the workers disqualified are discussed in counterpoint to the own developers' statements. We consider the hypothesis of taylorization inaccurate to describe the work experience of our interviewees; since the assumptions of the Taylorist organization do not dictate the division of production of software factories, it would be more appropriate to reflect on the process of rationalization of software production.

Keywords: ***Work; Software developers; Software Factory; Taylorization***

Introdução

A atividade de desenvolvimento de software surge num ambiente de produção, poderíamos dizer, artesanal. Giuseppe Cocco e Gilvan Vilarim (2009) narram a história do software partindo deste momento de grande personalização dos produtos (cada programador imprimia no software fortes traços subjetivos e criativos na codificação de instruções à máquina), até o momento, caracterizado por uma abordagem industrial da produção de software. Essa industrialização é evidenciada pela disciplinarização da programação, com o esforço do capital em suprir os meios de produção, deixando ao trabalhador o emprego da força de trabalho. O processo produtivo passou a ser segmentado e, consequentemente, foi criada uma hierarquia vertical dos profissionais que se encaixam nas diversas fases de elaboração do software.

É diante dessa conformação, e das promessas alardeadas por análises sociológicas como o informacionalismo de Manuel Castells (1999) e Jean Lojkine (1995) e o capitalismo cognitivo de Carlo Vercellone (2009), que surge o argumento da infotaylorização. Para Ruy Braga (2009), categorizar o trabalho imaterial como infotaylorismo tem a finalidade de contrapor tais promessas[1] à realidade degradada de trabalhadores da indústria de Tecnologias da Informação e Comunicação (TIC).

Este trabalho visa debater, a partir da aproximação empírica da realidade dos desenvolvedores de software, a hipótese da taylorização do trabalho no ambiente das fábricas de software. O argumento que queremos apresentar é de que a realidade desses trabalhadores é apenas parcialmente marcada pelos pressupostos da Teoria da Administração Científica, a saber: individualização e decomposição do trabalho; descrição pormenorizada dos postos de trabalho; planejamento minucioso da produção; distinção clara entre planejamento e execução (Silva, 2004). Sob o risco de nos opormos a um argumento repleto de conotação política, o do infotaylorismo, queremos, em verdade, apresentar um modo mais fidedigno de apreensão da realidade de trabalho dos desenvolvedores de software, visando, igualmente, fazer coro àqueles que produzem um discurso de crítica ao capitalismo tardio e seus mecanismos de subsunção do trabalho.

O setor de produção de software é marcado pela heterogeneidade. Há uma diversidade de produtos e serviços comercializados, de modelos de contratação de mão de obra e de organização do trabalho. Roger Pressman (2011) delimita sete grandes categorias de aplicação de software: softwares de sistema — programas feitos para atender a outros programas, comumente estão em comunicação com a parte física dos dispositivos, o hardware; softwares de aplicação — criam soluções para necessidades específicas de um negócio, como operações comerciais ou tomadas de decisão administrativas e técnicas; softwares científicos ou de engenharia — programas que permitem a análise de dados de pesquisa e a simulação de ambientes em diversas áreas da ciência, ou a fabricação automatizada de um dado bem; softwares embutidos — residem nos produtos ou sistemas que integram e têm a finalidade de prover funções ao usuário final, por exemplo, os programas presentes em automóveis e fornos micro-ondas; softwares para linha de produtos — criados para prover funções específicas a usuários diferentes, como os editores de textos e planilhas eletrônicas; aplicações para Web — programas voltados para funcionamento na internet, que abrangem uma amplitude de finalidades; softwares de inteligência artificial — aplicações à robótica e ao reconhecimento de padrões de imagem e voz fazem parte desta categoria. Por vezes, a comercialização de software resulta na prestação de serviços, de consultorias à manutenção e suporte ao usuário.

As formas de contratação no setor são, também, variadas. Maria Aparecida Bridi e Mariana Braunert (2015), a partir de pesquisa realizada com trabalhadores e empresários do setor de software em Curitiba e Região — PR, observaram quatro modos distintos de relação de trabalho: CLT — trabalhadores amparados pela Consolidação das Leis do Trabalho; PJ (pessoa jurídica), desenvolvedores que prestam serviço como microempresa ou microempreendedor individual; cooperados — trabalhadores contratados via cooperativas profissionais; "CLT Flex", trabalhadores que têm apenas parte da remuneração declarada na carteira de trabalho, e recebem o restante do

[1] Entre elas a esperança de que o trabalho imaterial, vivo e colaborativo em sua definição, enseje a superação da degradação do trabalho e o surgimento de uma economia da abundância.

salário "por fora". Os desenvolvedores contratados segundo a CLT gozam dos direitos e garantias previstas pelo direito do trabalho neste marco legal[2]. Os PJ's experimentam uma relação particular de prestação de serviços, já que se responsabilizam pela formalização burocrática de sua "empresa" e devem buscar as garantias relacionadas à previdência e outros cuidados que os assegurem diante dos imprevistos por si mesmos, permanecendo, contudo, em uma relação de subordinação que em nada difere da de outros trabalhadores. A tendência ao pejotismo marca a experiência laboral dos desenvolvedores e pode ser compreendida como estratégia das empresas de software para driblar os gastos relacionados à proteção social ligada ao trabalho, o que garante a contratação a custos mais baixos (Bridi & Braunert, 2015). A contratação via cooperativas profissionais segue propósito semelhante ao do pejotismo, no caso, são utilizadas "cooperativas de fachada", que medeiam uma relação de trabalho inscrita fora do que preconiza a CLT. A "CLT Flex" indica mais fortemente o impulso por burlar os encargos trabalhistas e assumir relações que se inscrevem no âmbito da ilegalidade, com o propósito de conquistar melhores condições de concorrência diante de um mercado globalizado.

Sobre os modos de organização do trabalho, é possível encontrar desde empresas regidas como fábricas de software, pautadas em modelos de desenvolvimento que buscam a racionalização do processo produtivo, a partir da segmentação e do uso de normas de qualidade, a empresas baseadas em metodologias ágeis de desenvolvimento, constituídas por pequenas equipes multifuncionais cuja produção é baseada em processos iterativos.

A pesquisa que subsidia este artigo foi realizada entre desenvolvedores de software de aplicação (aqueles voltados às necessidades de negócio específicas de uma empresa-cliente) das fábricas de software da cidade de Fortaleza – CE, que atuavam segundo modelos organizacionais os mais diversos e tendo trajetórias profissionais marcadas por experiências diversas de contratação. O grupo investigado é composto por 6 desenvolvedores com no mínimo 1 ano de experiência em fábricas de software. Todos os entrevistados são do sexo masculino e a média de idade da amostra é de 30.8 anos. Foram realizadas entrevistas semiestruturadas que tiveram como disparador a história laboral destes desenvolvedores. Os resultados que discutimos neste documento não permitem uma generalização acerca da experiência dos desenvolvedores de software, mas, trazem luzes sobre o modo como estes vivenciam as tendências de flexibilização e racionalização do trabalho no setor.

A discussão que empreendemos se desenrola a partir de uma triangulação entre as análises teóricas dessa atividade, o discurso dos desenvolvedores entrevistados e a interpretação que lhe damos. A opção se justifica pela abordagem sóciohermenêutica da análise do discurso (Alonso, 1998; Cárcamo, 2005; Ruiz Ruiz, 2009), metodologia que propõe, como o fazer do cientista social, a interpretação para além do sentido denotativo da linguagem. Para Luís Enrique Alonso (1998) todo discurso remete à pluralidade, pois permite uma diversidade de associações; remete também à instabilidade, visto que nenhum discurso está esgotado em suas definições; e à indexicalidade, conceito-chave da etnometodologia utilizado para exprimir como o significado de uma expressão é sempre dependente do contexto de sua enunciação (Iñiguez, 2004).

A seguir, discutiremos a hipótese de taylorização do desenvolvimento de software a partir das características que apontam a desqualificação técnica dos desenvolvedores, as estratégias de controle e disciplinares empregadas na organização do trabalho, e a possibilidade de socialização da produção mediante o uso das normas de qualidade, sempre referenciando o discurso dos trabalhadores investigados como em um diálogo provocado de fora. Ao final do artigo, desejamos reposicionar o debate sob o conceito de racionalização do trabalho e apontar possibilidades de investigação que permitiriam a ampliação da análise sobre a atividade de desenvolvimento de software.

[2] Com as recentes mudanças na legislação trabalhista brasileira (sancionadas em julho de 2017), posteriores à análise de Maria Aparecida Bridi e Benilde Lenzi Motim (2015), deve-se reconhecer que parte desses direitos e garantias já não estão mais vinculados à CLT.

A hipótese da taylorização da atividade de desenvolvimento de software

Na literatura disponível acerca do trabalho dos desenvolvedores de software, as tentativas de disciplinarização da produção são, comumente, atribuídas às normas de qualidade, como as normas ISO (*International Organization for Standardization*) e o modelo CMMI (*Capability Maturity Model Integration*), já que estas aumentariam o controle da gerência sobre os desenvolvedores. Além disso, a documentação dos processos adotados para a produção de softwares permitiria à gerência prescindir da capacidade criativa destes trabalhadores e tornaria menos custosa a substituição dos mesmos. Monica Prasad (1998) observa que, entre os anos de 1990 e 1994, mil subdivisões de empresas multinacionais na Índia solicitaram a certificação ISO, o que, para ela, mais do que padronizar a qualidade dos produtos e dos processos, funcionava como garantia para se vender software a compradores externos, como a União Europeia e os Estados Unidos, por exemplo, caracterizando-se, assim, como mecanismo de penetração em um mercado cada vez mais globalizado.

Dois dos excertos de entrevista com os desenvolvedores parecem corroborar a visão de Prasad (1998) sobre o uso das normas mais como um requisito para participação no mercado do que um guia para a organização de processos. No primeiro desses relatos, o entrevistado 1 (programador de software, CLT) afirma que a empresa em que trabalha passou recentemente pela auditoria para determinar o seu grau de maturidade CMMI. Ao ser questionado se antes da auditoria a empresa já utilizava as normas CMMI como guia da produção, o desenvolvedor explica: "É porque é muito relativo, assim, como é que eu explico, isso é mais uma coisa para você, sei lá, fazer uma propaganda internacional, no caso do CMMI, ou fazer uma propaganda nacional no caso do MPS.BR[3]" (Entrevistado Nº 1[4], entrevista pessoal, março de 2015). Já o entrevistado 3 (analista de sistemas, CLT) deixa transparecer certa irrelevância dos modelos em sua atividade concreta:

> A empresa que eu trabalho é uma empresa espanhola, tá? Ela é uma empresa muito organizada, muito grande. Tem atuações em vários mercados mundiais e, certamente, não tenho nem dúvida de que ela atenda a um desses padrões. Só não sei te dizer quais são. (Entrevistado Nº 3, entrevista pessoal, abril de 2015).

A fala do entrevistado 4 (programador de software, CLT), acerca da auditoria para aquisição do CMMI em sua empresa, faz refletir sobre processos de resistência dos trabalhadores, dentro da organização da qual faz parte, quanto às prescrições estabelecidas pelo modelo. Segundo o desenvolvedor, os gestores da fábrica de software enviavam por e-mail um guia de entrevista já respondido, buscando antecipar as perguntas que os trabalhadores poderiam responder aos auditores. Nas palavras dele: "Então eles já te mandavam isso pronto, por e-mail, você lia, decorava e quando o auditor lhe perguntava isso você respondia com aquilo que você tinha decorado, não necessariamente o que acontecia" (Entrevistado Nº 4, entrevista pessoal, março de 2015). O modelo de qualidade se impõe como uma prescrição temporária, delimitada pela presença da empresa auditora na fábrica de software, com a qual o desenvolvedor afirma lidar como faria com uma "prova escolar". Após a saída dos auditores, a produção volta a funcionar segundo as negociações e equilibrações próprias da cultura organizacional da fábrica.

Prasad (1998) afirma que a organização de processos a partir de modelos de qualidade contribui para a taylorização da produção de software, uma vez que as técnicas de documentação impostas liberam os postos de trabalho concreto, resultando em uma dinâmica de desqualificação da atividade.

É possível apontar no discurso dos desenvolvedores percepções que corroboram a ideia de desqualificação da atividade. Como, por exemplo, a fala do entrevistado 2 (líder técnico, CLT), "a grande maioria [dos desenvolvedores], infelizmente, só quer sentar numa cadeira, cuspir código e ganhar dinheiro por isso. Existe um termo técnico para essas pessoas: é macaco programador. Vem do inglês *code monkey*" (Entrevistado Nº 2, entrevista pessoal, março de 2015).

[3] Programa de Melhoria do Processo de Software Brasileiro, proposto pelo Governo Federal em 2003, é uma certificação que se propõe mais acessível para empresas brasileiras.

[4] Com a finalidade de manter o anonimato dos trabalhadores entrevistados, seus nomes foram suprimidos e utilizamos números para nos referirmos a eles.

A possibilidade de que um trabalhador imaterial como o desenvolvedor, que depende majoritariamente de suas faculdades cognitivas para realizar sua atividade, seja comparado a um animal como o macaco, indicaria uma simplificação extrema das tarefas, mesmo um esvaziamento do esforço cognitivo delas, o que deporia em favor do argumento da desqualificação. A atividade torna-se tão simples que qualquer profissional subqualificado pode exercê-la. O fato de que quatro dos seis desenvolvedores entrevistados ingressaram em fábricas de software sem ter concluído curso superior na área, apenas com cursos técnicos ou certificações em ferramentas específicas, apontaria para uma tendência de mercado de selecionar trabalhadores com uma qualificação restrita.

A hipótese da desqualificação é repelida por Vigneswara Ilavarasan e Arun Kumar Sharma (2003), ao afirmarem que, conforme as fábricas de software indianas (seu objeto de estudo) aumentam sua capacidade de executar projetos mais complexos, o trabalho com desenvolvimento ganha em criatividade. Os autores levantam seis hipóteses que podem ser aplicadas ao estudo de qualquer produção de software para se investigar se há rotinização do processo. São elas:

> 1. Os profissionais de software estão claramente divididos entre trabalhadores de concepção e de execução, tais como designers, codificadores e executores de testes;
>
> 2. Os trabalhadores de execução não participam da parte de concepção do projeto;
>
> 3. Os trabalhadores implicados em um módulo não têm conhecimento dos outros módulos no mesmo projeto;
>
> 4. Os requisitos de formação são diferentes para as distintas categorias de trabalhadores;
>
> 5. As oportunidades de carreira são restritas para os trabalhadores de execução;
>
> 6. Os procedimentos de certificação potencializam o controle gerencial (Ilavarasan & Sharma, 2003, p. 2, tradução livre).

A partir da pesquisa empírica com entrevistas semiestruturadas e observação direta de trabalhadores de duas fábricas de software de Bangalore, Ilavarasan e Sharma (2003) negam a proposição de que o trabalho com software esteja rotinizado e afirmam ser muito difícil de acontecer, dada a versatilidade da própria tarefa.

Os discursos colhidos entre nossos entrevistados mostram ser possível responder afirmativamente algumas dessas questões se aplicadas às suas realidades. Que há, sim, divisão no processo de desenvolvimento entre concepção e execução, diante do estabelecimento de uma hierarquia que aborda o ciclo de vida do software, desde a coleta de requisitos com o cliente à aplicação e manutenção do produto, e que os trabalhadores da execução são excluídos do processo de concepção do projeto (hipóteses 1 e 2 de Ilavarasan e Sharma, 2003). A fala do entrevistado 4 (programador de software, CLT) apresenta esse dado de forma minuciosa:

> O software passa por uma concepção; depois ele faz um projeto; depois ele passa para os analistas; os analistas pensam num software, eles projetam a coisa toda, passam para os programadores; os programadores desenvolvem o que os analistas projetaram; e depois passa para testes; manutenção e depois volta para o começo. Concepção de novas coisas e por aí vai. Então, qual o problema disso? Quando você está numa parte lá de desenvolvimento, tu não tem que pensar nada. O analista, ele já pensou as coisas por você. Já projetou aquilo ali. Então, na verdade, você recebe uma resma de papel na tua mesa, descrito tudo como tem que funcionar e como você vai fazer, e você é, basicamente, um digitador. Você vai sentar lá e desenvolver o que aquele cara projetou. Em algumas linguagens, inclusive, o cara já consegue projetar dentro da linguagem o que você tem que fazer. Então, ele te dá só o esqueleto e você preenche só o que tem dentro do esqueleto. (Entrevistado Nº 4, entrevista pessoal, março de 2015).

Também é possível responder afirmativamente quanto ao desconhecimento de outros módulos dentro do projeto (hipótese 3 de Ilavarasan e Sharma, 2003). O entrevistado 4 (programador de software, CLT) afirma que é possível, como já lhe aconteceu, produzir um segmento de código sem jamais chegar a saber a que porção específica do sistema este será acrescentado ou mesmo sua finalidade.

É possível, também, observar que há necessidades de formação diferentes entre trabalhadores de execução e de planejamento (hipótese 4 de Ilavarasan e Sharma, 2003), não sendo exigido destes que saibam a programação em si, nem daqueles que compreendam o modelo de negócios do cliente ou que possam propor soluções para o mesmo.

As possibilidades de carreira, (hipótese 5 da lista apresentada), foi tema bastante abordado pelos desenvolvedores entrevistados. A percepção das possibilidades de ascensão é

bastante distinta entre as faixas etárias. Algo próximo ao constatado por Maria Aparecida Bridi e Benilde Motim (2014) em pesquisa com 30 estudantes do curso de Tecnologia em Análise e Desenvolvimento de Sistemas de uma universidade pública do Paraná, que também trabalhavam ou estagiavam no setor.

Os estudantes-trabalhadores mais jovens valorizavam a flexibilidade e a rotatividade no trabalho como meio de obter experiências e encarar novos desafios, o que justificava a preferência por trabalhos flexíveis e instáveis em detrimento de estabilidade de carreira numa empresa com contrato por tempo indeterminado. Os dois trabalhadores mais jovens, dentre o grupo investigado, parecem ilustrar o dado apresentado por Bridi e Motim (2014).

O entrevistado 5 (programador de software, CLT), 23 anos, enaltece a natureza adaptável dos profissionais da área de programação, formados em um ambiente de trabalho bastante dinâmico e, por isso, capazes de assimilar com facilidade as demandas de setores diversos, tais como o financeiro ou de pessoal. O entrevistado 1 (programador de software, CLT), 24 anos, ressalta experimentar esse momento de adaptação em sua carreira, já que está em vias de deixar a atividade de consultor para atuar como líder de uma equipe dentro de sua empresa.

As respostas avaliadas por Bridi e Motim (2014) dos menos jovens mostram preocupações com estabilidade, com ganhar mais dinheiro e alcançar benefícios, direitos e garantias relacionados ao trabalho — um meio buscado, para tanto, são os concursos públicos. Entre eles, a percepção é de obstáculos bem maiores à ascensão laboral. O entrevistado 6 (programador de software, desempregado), 32 anos, viu suas possibilidades de promoção condicionadas pela lucratividade gerada pelos projetos dos quais participava. Aos poucos foi percebendo que as promessas de crescimento dentro da organização não eram tão simples como a gerência fazia parecer.

Já o entrevistado 3 (analista de sistemas, CLT), 48 anos, o mais experiente entre os desenvolvedores do grupo investigado, percebe o mercado com características bem diferentes daquelas que observou no período de seu ingresso. Segundo ele, com pouco tempo de atuação os desenvolvedores atingem o salário limite e ficam sem perspectiva de ascensão.

De acordo com Bridi e Motim (2014), a necessidade de conhecimento não somente teórico como prático para atuar no setor de informática tende a atrair jovens iniciantes. No Brasil, o setor de Economia da Informação apresenta os trabalhadores com idade média mais baixa, 32 anos (Softex, 2013). As autoras, partindo dessa constatação e da valorização do par flexibilidade-estabilidade, dividem a carreira no setor de TIC em dois momentos principais: no início, possivelmente com a preocupação de adquirir experiência, os trabalhadores são empregados informais ou estagiários; no segundo momento, com alguns anos de experiência e certa autonomia, é que se busca a efetivação através do emprego formal ou mesmo da prestação de serviço como pessoa jurídica (Bridi & Motim, 2014).

O relato do entrevistado 6 (programador de software, desempregado) corrobora a ideia de que experiência e autonomia são acompanhadas pelo desejo de estabelecer-se por conta própria, associado à preocupação com um futuro estável.

> Eu sempre tive a vontade de fazer algo, um sisteminha, alguma coisa, que eu possa ter uma renda dele sem trabalhar. E hoje em dia, eu trabalhando por conta própria, com meus projetos todos, têm a possibilidade de fazer isso. De ir tocando em paralelo uma coisa que no futuro vai me dar, vai me garantir uma renda extra, algo que eu não precise me preocupar tanto. (Entrevistado Nº 6, entrevista pessoal, abril de 2015).

Já o relato do entrevistado 3 (analista de sistemas, CLT) revela um desencanto com a própria carreira no setor de TIC relacionado às dificuldades de crescimento profissional. Em sua fala, a ambivalência entre adorar o que faz e o desejo de mudar de profissão é resolvida em favor da permanência na área, justamente considerando-se o fator idade: "se eu pudesse mudar hoje, eu voltaria para a Engenharia Elétrica. Não estaria mais na área de TI. Entendeu? Só que eu, já devido à idade [48 anos], devido ao tempo… é uma mudança muito complicada, muito delicada" (Entrevistado Nº 3, entrevista pessoal, abril de 2015).

Para dar continuidade ao diálogo com Ilavarasan e Sharma (2003), e discutir a última da sua lista de hipóteses, "os procedimentos de certificação potencializam o controle gerencial", daremos atenção particular à discussão sobre os mecanismos de controle presentes no processo de software.

Guillermina Yansen, Lucila Dughera, Nahuel Mura e Mariano Zukerfeld (2012) refletem sobre as relações de poder na produção de software, a partir de trabalhadores e empresas da cidade de Buenos Aires, e afirmam que tais relações são caracterizadas por um relaxamento dos mecanismos disciplinares em favor de mecanismos de controle[5]. Observam as relações de poder no âmbito da produção de software com base em cinco eixos, a saber: a gestão do tempo; o nível de predeterminação das tarefas; os métodos de avaliação aplicados; o espaço arquitetônico em que se desenvolve a atividade; e a vestimenta/aparência dos desenvolvedores de software.

A gestão do tempo nas empresas de software observadas por Yansen et al. (2012) é flexível, por mais que quase todos os entrevistados tenham horário fixo de jornada, não o cumprem de maneira estrita. Um gerente afirma que não controla os horários, mas os resultados. Chamou a atenção dos autores um tipo particular de regulação: programas de monitoramento que acompanham o uso do tempo ao longo da jornada dos trabalhadores.

Um primeiro tipo de software fraciona os tempos dedicados a tarefas específicas e gera estatísticas que, em última instância, permitem o cálculo de custos em um determinado projeto. Um segundo tipo de programa monitora todas as ações do trabalhador no computador, levanta estatísticas sobre a relação tempo/tarefa, permite estipular objetivos e etiquetar atividades de acordo com seu grau de dificuldade. E mais, o programa adverte o trabalhador caso esteja usando tempo além do permitido com distrações como redes sociais, por exemplo. Os autores citam dois programas como exemplo, o *Intranet Labour Claiming* da IBM no primeiro caso, e o *Rescue Time* no segundo.

A partir disso, Yansen et al. (2012) apontam que "o primeiro destes softwares parece favorecer o controle; o segundo, a disciplina. Porém, é interessante que nenhum deles permite identificar e eliminar 'os tempos mortos', objetivo da organização disciplinar" (p. 79, tradução livre). Para eles, no que concerne à gestão do tempo, na produção de software predominam os mecanismos do controle sobre os disciplinares.

O entrevistado 2 (líder técnico, CLT) descreve uma dessas ferramentas em sua experiência:

> É uma ferramenta de controle de ideias, de *braainstorm*, de requisitos. Lá eles chamam de *loops*. […] Eles [os clientes] criam lá um *loop* que tem todas as informações do que tem que ser feito. E os *loops* vão sendo atualizados muito rápido. E com bastante frequência. Então, é como se fosse um espaço…tem características de fórum e de rede social. […] E aí todo dia, a gente lança pelo menos um registro cada um [dos integrantes da equipe] nessa ferramenta dizendo o que a gente fez. (Entrevistado Nº 2, entrevista pessoal, março de 2015).

No que se refere ao nível de predeterminação das atividades, Yansen et al. (2012) encontram grande variação entre as respostas obtidas. Há um primeiro grupo que tem maior parte de sua jornada, entre 70% e 80%, ocupada com tarefas predeterminadas; fazem parte desse grupo programadores, administradores de bancos de dados e, em alguns casos, sócios de microempresas. O segundo grupo, formado por profissionais de gerência, afastados da atividade de programação em si e dedicados a atividades de supervisão e coordenação de projetos, afirmam ter a quase totalidade de suas atividades pautadas. O terceiro grupo experimenta níveis mínimos de predeterminação, formado por donos de microempresas, investigadores de tecnologia em empresas de grande porte e multinacionais, e produtores de software livre. Essa heterogeneidade de experiências leva os autores a concluírem que:

> A ideia de que os trabalhadores informacionais trabalham em condições de alta criatividade e baixos níveis de rotinização só se verifica em alguns processos produtivos. Isto não quer dizer que se mantenha linearmente a lógica disciplinar do trabalho industrial, mas que se deve evitar a crença automática de que se trata de atividades sempre novas. (Yansen et al., 2012, p. 80, tradução livre).

Quanto aos métodos de avaliação, Yansen et al. (2012) observam entre o grupo investigado modelos de avaliação por supervisores, líderes de equipe e gerentes, que estabelecem pontuações sobre o comportamento global dos trabalhadores. Ao fim de períodos semestrais

[5] Yansen et al. (2012) reconhecem a distinção entre esses mecanismos a partir dos estudos de Foucault e de Deleuze sobre as relações de poder. Os mecanismos disciplinares, característicos do capitalismo industrial, se baseiam na vigilância minuciosa das operações do corpo com a finalidade de gerar indivíduos dóceis e úteis. Já os mecanismos de controle, associados ao capitalismo informacional, estão baseados na regulação da diferença, cujo potencial deve ser posto a serviço da ordem vigente. Os mecanismos de controle se aplicam não sobre o corpo do indivíduo mas sobre o corpo social mais extenso.

ou anuais, os trabalhadores se encontram com os gestores que assinalam os escores de avaliação. Todavia, os trabalhadores encontram dificuldades em descrever os princípios desses modelos, o que demonstra que, por mais que todos tenham consciência de sua existência, o modo como são realizadas as avaliações é um tanto nebuloso, ficando os trabalhadores alheios aos critérios específicos e às consequências das avaliações.

Apesar de existirem e de os trabalhadores terem consciência delas, não são as estratégias de avaliação que pautam o comportamento dos trabalhadores, o que levaria a conclusão de que "este sistema atua de forma flexível e não como um marco rígido de normas segundo as quais estão predeterminadas as consequências para cada qualificação" (Yansen et al. 2012, p. 81, tradução livre). A ausência de um sistema de punições e recompensas, a tendência a avaliar períodos extensos de tempo e a presença de bonificações outorgadas de maneira automática são características que afastam esses métodos de avaliação do paradigma disciplinar.

A arquitetura dos espaços de trabalho é formada por ambientes com plantas abertas e escritórios organizados de maneira anárquica que privilegiam a interação entre os trabalhadores. Em algumas empresas multinacionais investigadas, Yansen et al. (2012) relatam a existência de uma área comum com videogames, fliperama, geladeira e máquinas de guloseimas, de livre acesso para os trabalhadores. Nessas mesmas empresas é possível observar, também, espaços proibidos aos trabalhadores. Para os autores, definir a arquitetura como mecanismo disciplinar ou de controle se torna um desafio. Se de um lado a existência de uma área comum, que borra as fronteiras entre tempo de trabalho e lazer, remete à lógica dos mecanismos de controle, de outro, a existência de áreas proibidas traz à lembrança mecanismos disciplinares.

A aparência dos trabalhadores é a dimensão em que o relaxamento disciplinar descrito pelos autores se torna mais patente. Não há códigos de vestimenta nem padrões como uniformes que diferenciem os empregados segundo seu lugar na hierarquia da empresa. Os desenvolvedores se vestem de maneira bastante informal, com roupas desgastadas e não engomadas, algo bastante diferente do código de conduta dos trabalhadores de "colarinho branco". Outro fato que chama a atenção é que não há diferença na maneira de se vestir dentro e fora do ambiente de trabalho, o que ajuda a borrar um pouco mais a fronteira entre tempo de trabalho e lazer.

Ao fim da investigação, Yansen et al. (2012) recolocam a questão das relações de poder na produção de software pautadas pela lógica disciplinar ou pela lógica do controle, e concluem que, dada a heterogeneidade observada em torno dos eixos temáticos analisados, existe retraimento da aplicação de mecanismos disciplinares na produção de software, se comparada com a produção no período industrial; prevalência do uso de mecanismos de controle; e variação do uso de cada um desses mecanismos, segundo o tipo de processo produtivo e o papel desempenhado pelo trabalhador.

No estudo das relações de poder no trabalho, o modelo taylorista (assim como o modelo de organização fordista) está associado mais diretamente com os dispositivos do poder disciplinar, servindo, inclusive, o espaço da organização taylorista como locus de observação desses dispositivos. A lógica do controle, traço da passagem a uma sociedade pós-industrial ou pós-moderna, está mais claramente identificada com a lógica de produção toyotista.

Recorrer aos mecanismos do controle para organizar a produção indica um modo subreptício de ordenar, organizar e envolver os sujeitos no trabalho. O reconhecimento desses novos parâmetros das relações de poder no ambiente do trabalho imaterial exige que matizemos a hipótese de Ilhavarasan e Sharma (2003), "os procedimentos de certificação potencializam o controle gerencial". A princípio, é possível, como veremos a seguir na fala do entrevistado 4 (programador de software, CLT) sobre o CMMI e a predeterminação de tarefas, considerar que a hipótese se confirma, contudo, seu relato constitui a menor frequência nos discursos do grupo investigado.

> O CMMI era um modelo que eu não gostava muito porque ele tirava do desenvolvedor toda a propriedade do projeto. Como é que funciona isso? Ele, não necessariamente, mas na maioria das vezes, ele segue o modelo cascata. O que é um modelo cascata? O software passa por uma concepção, depois ele faz um projeto, depois ele passa para os analistas. Os analistas pensam num software, eles projetam a coisa toda, passam para os programadores. Os programadores desenvolvem o

> que os analistas projetaram, e depois passa para testes, manutenção e depois volta para o começo. Concepção de novas coisas e por aí vai. Então, qual o problema disso? Quando você está numa parte lá de desenvolvimento, tu não tem que pensar nada. O analista, ele já pensou as coisas por você. Já projetou aquilo ali. Então, na verdade, você recebe uma resma de papel na tua mesa, descrito tudo como tem que funcionar e como você vai fazer, e você é basicamente um digitador. Você vai sentar lá e desenvolver o que aquele cara projetou. Em algumas linguagens, inclusive, o cara já consegue projetar dentro da linguagem o que você tem que fazer. Então, ele te dá só o esqueleto e você preenche só o que tem dentro do esqueleto. (Entrevistado Nº 4, entrevista pessoal, março de 2015).

O tom da grande maioria dos relatos que colhemos é muito mais aproximado da lógica descrita por Yansen et al. (2012) como dos mecanismos de controle. Os trabalhadores têm horários flexíveis, podem continuar parte do trabalho de casa, as tarefas são diversificadas e pouco predeterminadas, não usam fardas ou uniformes e nem há normas quanto a cortes de cabelo e barba, e algumas empresas possuem área comum com lanches e jogos.

Para Carmem Grisci (2008) os mecanismos do controle são respostas do capital ao desafio de engajar aspectos tão subjetivos como criatividade e autonomia dentro da produção. Não há outro meio senão pela mobilização psíquica, fazendo os sujeitos controlarem a si mesmos. A preocupação do entrevistado 2 (líder técnico, CLT), quanto aos ciclos de entregas de códigos que este estabeleceu para si e para sua equipe, ilustra a maneira como a mobilização psíquica faz com que o trabalhador se responsabilize pelo controle da própria produtividade.

> Então todo dia, o que a gente tiver feito, a gente joga lá [no servidor]. A gente pega o apurado de todo dia, tudo que a gente fez. Eu não gosto de dias em que a gente chega para o cliente e diz "olha, hoje eu não tenho nada para mostrar". A gente pega o que a gente fez. Tá no servidor de desenvolvimento, o cliente tem um servidor de testes e produção. Então, a gente pega tudo que a gente tem feito e estabilizado, por assim dizer, vai para o servidor de testes. O cliente tem como ver. Ele vê, todos os dias ele vê mudanças acontecendo no sistema dele. (Entrevistado Nº 2, entrevista pessoal, março de 2015).

Ilavarasan e Sharma (2003) concluem, contrários a Prasad (1998), que é improvável que as normas de qualidade resultem em incremento do controle gerencial, já que os procedimentos de documentação beneficiam os desenvolvedores, aumentando o controle destes sobre a própria atividade. Além disso, os autores reconhecem na realidade indiana o mesmo que apontam os relatos dos entrevistados 1 (programador de software, CLT) e 3 (analista de sistemas, CLT) no início deste tópico, que as certificações são utilizadas mais como um expediente para atrair clientes do que um referencial de organização da produção.

Um questionamento que acrescentaríamos à lista de Ilavarasan e Sharma (2003) sobre a rotinização do processo de software seria: se a atividade se torna repetitiva, poderíamos tratar de um esvaziamento do sentido da atividade? Nesta perspectiva, ao interpelarmos o entrevistado 4 (programador de software, CLT) sobre essa repetitividade, obtivemos a seguinte resposta:

> Ele [o trabalho] não se torna repetitivo necessariamente. Dentro do mesmo projeto, às vezes, ele acaba sendo, mas, depois de um tempo de trabalho, quando você muda de projeto, acaba mudando um pouco de história, não é? É muito diferente você trabalhar para um projeto de banco, para um projeto de software de celular. As maneiras, as coisas que você faz se diferenciam um pouco, mas, de novo, você não pensa naquilo. (Entrevistado Nº 4, entrevista pessoal, março de 2015).

O relato revela em que termos há diversidade de tarefas na atividade dos desenvolvedores. A produção de softwares de aplicação, segmento em que se encontram todos os desenvolvedores do grupo investigado, tem por característica o uso da computação como meio de gerar soluções para necessidades de negócio específicas. A alocação em projetos diferentes exige que esses trabalhadores se relacionem com novos requisitos, formam-se novas equipes e, por vezes, é necessário aprender novas linguagens e ferramentas.

Paul Adler (2004) afirma, a partir de investigações com quatro grandes empresas estadunidenses, acontecer nas fábricas de software sob a organização do CMM um processo de socialização da produção, que pode ser descrito em seis índices: a) ampliação do objeto de trabalho, que implica dizer que quanto mais avançado o CMM da organização, mais os programadores compreendem o que os outros trabalhadores estão fazendo e o porquê, amenizando a separação entre executores e designers; b) ampliação do trabalhador coletivo, relacionada à necessidade de cooperação entre grupos e consulta aos pares, por

exemplo; c) aprofundamento da interdependência colaborativa, expresso no reconhecimento de que o esforço de cada um é apenas uma parte de um processo maior; d) socialização das ferramentas utilizadas no trabalho, fruto da preocupação dos desenvolvedores em entregar produtos de qualidade; e) socialização do desenvolvimento de regras e ferramentas, o que permite engajar os trabalhadores a partir da definição dos projetos; f) socialização dos processos de formação e qualificação (Adler, 2004).

Para Adler (2004) a socialização da produção está relacionada com um incremento da qualificação dos trabalhadores. Qualificação compreendida como "domínio da complexidade das tarefas exigidas dos trabalhadores em seus empregos, e domínio das relações que coordenam a atividade através destas tarefas" (Adler, 2004, pp. 3-4, tradução livre). Sua conclusão é oposta à de Prasad (1998), e Juan José Castillo (2009) afirma não encontrar, a não ser excepcionalmente, essa melhora da qualificação da atividade dos desenvolvedores em sua pesquisa de campo com os profissionais do norte de Madrid.

Quanto à nossa observação, retornamos ao relato do entrevistado 4 (programador de software, CLT), que descreve a maneira como a predeterminação da atividade pode tolher o poder de decisão do desenvolvedor, sem, necessariamente, impor-lhe uma repetitividade tediosa ou privar-lhe do uso da criatividade.

> Você simplesmente imagina o melhor código para funcionar dentro daquilo ali que alguém já pensou. Você imagina como aquilo ali vai rodar o mais rápido possível e ter uma eficiência grande, mas você não chega a […] Imagina que alguém chegasse para você e dissesse "eu quero a Monalisa desse jeito". Aí você olha aquilo ali e fala "Monalisa? Beleza! ". Tu vai olhar aquele quadro e vai buscar a maior perfeição na hora de pintar, escolher cores, etc. Mas se você quiser que a Monalisa esteja triste porque você acredita que aquilo é melhor, você não vai poder fazer. (Entrevistado Nº 4, entrevista pessoal, março de 2015).

Na realidade descrita os caminhos da socialização da produção parecem barrados, não restando ao desenvolvedor, dentro do processo, nada que não "tirar código" (Castillo, 2009). No relato do entrevistado 6 (programador de software, desempregado), a falta de socialização permite a imposição de estimativas irreais que geram sobrecarga de trabalho para o desenvolvedor.

> Quem estimava os projetos de desenvolvimento eram os consultores de 18, 19 anos de idade que não sabiam nem o que era um sistema. Então chegava lá coisa para tu fazer absurda. Tipo o fato de tu implementar um método estatístico da formiga num software para agendamento de obras de uma empresa gigante que atua no Brasil todo. Começa que você não entende nem de obra, nem de método estatístico e nem a tecnologia que vão usar para desenvolver o software. Aí você tem que aprender tudo isso em uma semana, entendeu? (Entrevistado Nº 6, entrevista pessoal, março de 2015).

Adler (2004) aponta contratendências dentro da organização baseada no CMM que podem distorcer a socialização no sentido da valorização do capital. As pressões por valorização do produto fazem com que os ganhos advindos da adoção do CMM, como melhora nos custos, na qualidade e nos prazos, deem lugar a prazos cada vez mais curtos e metas de lucro cada vez mais estreitas.

Reposicionando o debate sob o conceito de racionalização do trabalho

A administração científica do trabalho proposta por Taylor se desenvolve no bojo do espírito de racionalização da vida social que caracteriza o século XX. No entanto, alguns aspectos fundamentais da organização taylorista não são claramente identificados na produção dentro das fábricas de software.

Retomando o diálogo com Ilavarasan e Sharma (2003), há características particulares do processo de software que implicam barreiras à taylorização. A primazia do uso da cognição é ressaltada por eles, que afirmam haver espaço suficiente para que os desenvolvedores utilizem sua criatividade e sua imaginação no trabalho. Além disso, experiência e *expertise* não necessariamente caminham juntas na realidade desses profissionais, é possível que um desenvolvedor com dez anos de carreira possua pouquíssimo saber em dada linguagem, e conviva com colegas com bem menos experiência, porém especializados nessa mesma linguagem.

Castillo (2009) ressalta outra particularidade da produção de software: às duas opções disponíveis à produção fabril, produzir em casa ou comprar fora, na produção imaterial é acrescentada uma terceira, conectar-se. A opção por conectar-se tem por efeito a criação de equipes com formações, referências culturais e qualificações diversas, às vezes

constituídas por vínculos presenciais, às vezes por conexões virtuais. Para ele a análise da organização do processo de software deve partir da realidade dessas equipes, por ser nelas que reside o caráter realmente inovador desse mercado.

Castillo (2009) apresenta uma conclusão mais complexa do que a simples assunção da taylorização do trabalho dos desenvolvedores, nos seguintes termos:

> Muito bem, o aspecto antitaylorista, não individualizador e não rotinizável também estará sempre presente, ainda que possam se dar aspectos de intensificação do trabalho, de traslado da vigilância e disciplina no interior do grupo assim constituído e, portanto, aos próprios programadores. (Castillo, 2009, p. 35).

Reconhecemos também como característica particular da produção de software a interação entre as normas de qualidade e os modelos de processos, pensados pela Engenharia de Software. Na realidade nas fábricas de software a visão racionalista se expressa, antes das normas de qualidade, nos modelos de processos, que definem a divisão do trabalho dentro dessas empresas. Se as normas de qualidade dão ênfase aos passos da produção, como documentar e otimizar a produção de um software com vistas a atingir maior qualidade, os modelos de processos vão mais fundo, definindo o fluxo de produção, o escopo da atuação de cada profissional segundo sua especialização e a definição de tempos e metas. Assim, as normas de qualidade convivem com os modelos de processos dentro das empresas, por exemplo, uma fábrica organizada segundo o modelo *Rational Unified Process*[6] (RUP) que possui certificação CMMI.

Os modelos de processos geram uma separação funcional dentro do ambiente de produção, gerentes de projeto e de negócios estão em contato mais direto com o cliente e suas necessidades e têm maior responsabilidade de planejamento, já os analistas e programadores recebem as demandas e devem produzir requisitos e linhas de código de acordo com elas. Leandra Leal (2008) observa que este distanciamento das necessidades do cliente, por vezes, impossibilita o processo de aculturação que envolve a construção do software. Ela afirma que a produção de um software de aplicação envolve duas comunidades de saberes, uma constituída dentro da empresa pelos próprios técnicos, e outra pelo grupo de usuários. A qualidade do software depende da partilha de saberes entre estas duas comunidades, facilitada sobremaneira pelo contato direto entre elas.

> O trabalho do analista envolve várias pessoas (usuários leigos e profissionais de informática), implicando interações de diversas naturezas e em diversos momentos. O saber desse profissional faz parte de uma cultura e, para o aprendizado, é preciso que as pessoas compartilhem o mesmo ambiente, tornando a interação presencial entre os envolvidos imprescindível no momento de desenvolver soluções ou resolver problemas. (Leal, 2008, p. 35).

O argumento do infotaylorismo é produzido por Braga (2009) a partir da análise da atividade dos teleoperadores, os trabalhadores dos *call centers*, análise que, no Brasil, avançou quase ao ponto de um consenso. A aplicação do infotaylorismo não é adequadamente transferível ao desenvolvimento de software. As características fundamentais da organização taylorista são apenas parcialmente observáveis entre os desenvolvedores, e seus relatos apontam uma tendência de precarização desta atividade traduzida em modos de contratação cada vez mais flexíveis – sabemos que a flexibilização laboral é coetânea da precarização laboral, constituindo um par conceitual para a compreensão do mundo do trabalho –, no emprego de modelos de processos que dividem a produção entre planejamento e execução e na mobilização subjetiva destes trabalhadores.

Cabe compreender melhor os aspectos de precariedade presentes na atividade de desenvolvimento de software, já que ela é um exemplo da crescente "imaterialização" da economia contemporânea. Não queremos com esta comunicação nos opor simplesmente ao argumento do infotaylorismo, sobretudo por reconhecermos a força política que este possui, mas expandir o conhecimento acerca desta categoria de trabalhadores e de como se dá a precariedade em sua realidade. O caminho para novas pesquisas deve ser, ao nosso ver, pelo foco nas equipes e nos ciclos de comunicação que envolvem o processo de software.

[6] Abordagem industrial do desenvolvimento de software que propõe a divisão dos trabalhadores em analistas de sistema, desenvolvedores, testadores, trabalhadores de produção e suporte, e gerentes (Leal, 2008).

Referências

Adler, Paul (2004). Skill trends under capitalism and the socialization of production. In: Chris Warhust, Irena Grugulis & Ewart Keep (Eds.), *The Skills That Matter* (pp. 242-260), Houndmills: Macmillan Palgrave.

Alonso, Luís Enrique (1998). *La mirada cualitativa en sociología*. Madrid: Ed. Fundamentos.

Braga, Ruy (2009). A vingança de Braverman: o infotaylorismo como contratempo. *In*: Ricardo Antunes & Ruy Braga (Orgs.), *Infoproletários: degradação real do trabalho virtual* (pp. 59-88). São Paulo: Boitempo.

Bridi, Maria Aparecida & Braunert, Mariana Bettega (2015). O trabalho na indústria de software: a flexibilidade como padrão das formas de contratação. *Caderno CRH*, *28*(73), 199-213. https://doi.org/10.1590/s0103-49792015000100013

Bridi, Maria Aparecida & Motim, Benilde Lenzi (2014). Trabalho e trabalhadores na indústria de informática. *Contemporânea*. *4*(2), 351-380.

Cárcamo, Héctor (2005). Hermenéutica y análisis cualitativo. *Cinta Moebio*, *23*, 1-14. Retrieved from: http://www.redalyc.org/articulo.oa?id=10102306

Castells, Manuel (1999). *A sociedade em rede* (2. ed.) Rio de Janeiro: Paz e Terra.

Castillo, Juan José (2009). O trabalho do conhecimento na sociedade da informação: a análise dos programadores de software. *In*: Ricardo Antunes & Ruy Braga (Orgs.), *Infoproletários*: *degradação real do trabalho virtual* (pp. 15-36). São Paulo: Boitempo.

Cocco, Giuseppe; Vilarim, Gilvan de Oliveira (2009). Trabalho imaterial e produção de software no capitalismo cognitivo. Liinc em Revista, *5*(2), 173-190. https://doi.org/10.18225/liinc.v5i2.315

Grisci, Carmem Ligia (2008). Trabalho Imaterial, Controle Rizomático e Subjetividade no Novo Paradigma Tecnológico. *RAE Eletrônica*, *7*, 4, 1-23. https://doi.org/10.1590/s1676-56482008000100005

Ilavarasan, Vigneswara & Sharma, Arun Kumar (2003). Is software work routinized?: Some empirical observations from Indian software industry. *Journal of Systems and Software*, *66*(1), 1-6. https://doi.org/10.1016/S0164-1212(02)00050-X

Iñiguez, Lupicínio (2004). A linguagem nas ciências sociais: fundamentos, conceitos e modelos. In: Lupicínio Iñiguez (Coord.). *Manual de análise do discurso em ciências sociais* (pp. 50-104). Petrópolis: Vozes.

Leal, Leandra (2008). *Saber social e desenvolvimento de software: avaliação crítica do modelo da fábrica de software*. Dissertação de mestrado inédita. Universidade Federal de Minas Gerais.

Lojkine, Jean (1995). *A revolução informacional*. São Paulo: Cortez.

Prasad, Monica (1998). International Capital on "Silicon Plateau": Work and Control in India's Computer Industry. *Social Forces*, *77*(2), 429-452. https://doi.org/10.2307/3005534

Pressman, Roger S. (2011). *Engenharia de software: uma abordagem profissional*. *7*. Porto Alegre, RS: AMGH Ed.

Ruiz Ruiz, Jorge (2009). Análisis sociológico del discurso: métodos y lógicas. *Forum: Qualitative Social Research*, *10*(2), art. 26. Retrieved from: http://hdl.handle.net/10261/64955

Silva, Maria de Fátima S. (2004). A psicologia social e a psicologia (social) do trabalho. In: Maria de Fátima S. Silva & Cássio Adriano B. Aquino (Orgs.), *Psicologia social: desdobramentos e aplicações* (pp. 88-103). São Paulo: Escrituras Editora.

Softex (2013). *Cadernos Temáticos do Observatório: Economia da Informação e Internet. Campinas, SP: Associação para Promoção da Excelência do Software Brasileiro – SOFTEX.*

Vercellone, Carlo (2009). Lavoro, distribuzione del reddito e valore nel capitalismo cognitivo: una prospettiva storica e teorica. *Sociologia del Lavoro*, *115*, 31-54. https://doi.org/10.3280/SL2009-115002

Yansen, Guillermina; Dughera, Lucila; Mura, Nahuel & Zukerfeld, Mariano (2012). Mecanismos de poder en el trabajo informacional: la disciplina y el control en los procesos productivos de software. *Nómadas*, *36*, 75-89. Retrieved from: http://www.redalyc.org/articulo.oa?id=105124264006

DÍMITRE SAMPAIO MOITA

Doutorando do Programa de Pós-Graduação em Psicologia da Universidade Federal do Ceará. Psicólogo e Mestre em Psicologia pela Universidade Federal do Ceará.

CÁSSIO ADRIANO BRAZ DE AQUINO

Doutor em Psicologia Social pela Universidad Complutense de Madrid. Professor do Departamento de Psicologia da Universidade Federal do Ceará

VERÔNICA SIQUEIRA ARAÚJO

Mestre em Psicologia pela Universidade Federal do Ceará. Professora substituta do Departamento de Psicologia da Universidade Federal do Ceará

DIRECCIÓN DE CONTACTO

dimitremoita@gmail.com; brazdeaquino@gmail.com; araujo.siqueira.veronica@gmail.com

FORMATO DE CITACIÓN

Moita, Dímitre Sampaio; Aquino, Cássio Adriano Braz de & Araújo, Verônica Siqueira (2018).Diálogos sobre a taylorização da atividade dos desenvolvedores das fábricas de software. *Quaderns de Psicologia, 20*(2), 115-127. http://dx.doi.org/10.5565/rev/qpsicologia.1433

HISTORIA EDITORIAL

Recibido: 19-09-2017
Reevaluable: 06-04-2018
Aceptado: 12-06-2018