Desenvolvimento de software como desenho integrado de software e organização

João Porto de Albuquerque[1], Edouard Simon[2]

[1] Escola de Artes, Ciências e Humanidades, Universidade de São Paulo
Rua Arlindo Béttio, 1000 – Ermelino Matarazzo, São Paulo/SP
joao.porto@usp.br

[2] Departamento de Informática, Universidade de Hamburgo
simon@informatik.uni-hamburg.de

Abstract. *From a construction-oriented perspective, software development is closely tied with the intention of generalising organisational practices, abstracting them out of their particular original context. From the viewpoint of software use, in contrast, new organisational practices emerge from the employment of software artefacts in specific contexts. This essay strives to articulate these two viewpoints, understanding them within a cyclical process of organisational evolution, based on the one hand in the reflection of informatics researchers on the social aspects of software design, and on the other hand, in social theory that analyse the use of information systems. In this manner, this paper outlines a sociotechnical understanding of software development as integrated software and organisational design, thus emphasising the influence of the perceptions and interests of the involved actors, as well as their significance for the design process.*

Resumo. *De uma perspectiva orientada à construção, o desenvolvimento de software está diretamente relacionado à pretensão de generalizar práticas organizacionais, abstraindo-as de seu contexto particular original. Do ponto de vista do uso de software, porém, novas práticas organizacionais emergem a partir do emprego de artefatos de software em contextos específicos. Este ensaio procura articular essas duas perspectivas compreendendo-as dentro de um processo cíclico de evolução organizacional, apoiando-se, por um lado, na reflexão de pesquisadores de informática sobre os aspectos sociais do projeto de software e, por outro, em teorias sociais para análise do uso de sistemas de informação. Assim, o artigo apresenta os contornos de uma compreensão sociotécnica sobre o desenvolvimento de software como desenho integrado de software e organização, enfatizando a influência das percepções e interesses dos atores envolvidos, assim como seu significado para o processo de desenvolvimento.*

1. Introdução

A introdução de software tornou-se nas últimas décadas uma das formas mais pronunciadas de catalisar mudanças organizacionais. Processos de negócio complexos são hoje fortemente dependentes de software e novos softwares são freqüentemente introduzidos com o intuito de melhorar ou viabilizar novos procedimentos e novas

formas de organizar o trabalho. Como conseqüência dessa relação íntima entre software e organização, torna-se imperioso que o projeto e implantação de software leve em conta a estrutura pré-existente da organização – constituída historicamente por meio das contínuas ações dos atores em torno de seus processos, tarefas típicas, regras, instruções, formulários, hierarquias e rotinas – bem como avalie as conseqüências organizacionais resultantes da introdução do artefato de software. Para tanto, é necessária uma compreensão sociotécnica do processo de desenvolvimento de software que considere de maneira integrada a construção do artefato técnico e suas implicações organizacionais.

Na busca de tal compreensão, este ensaio toma por base pesquisas que abordam a mudança organizacional viabilizada por software a partir de duas perspectivas distintas. Primeiramente (Seção 2), examinamos o processo de *modelagem informática* proposto por Floyd (1997, 2002), que se aproxima das relações entre organização e software e a partir da perspectiva da construção deste. Em seguida, a Seção 3 inverte a perspectiva, analisando o processo de apropriação e uso do software do ponto de vista organizacional, como conceituado por Orlikowski (1992, 2000). Com base nessas duas perspectivas, a Seção 4 desenvolve uma visão do desenvolvimento de software como desenho integrado de software e organização, apresentando um processo cíclico de evolução organizacional composto de duas fases: a) a *descontextualização* de padrões de ação social; e b) a *recontextualização* de artefatos de software em um contexto organizacional. As considerações finais deste ensaio são expostas na Seção 5.

2. A perspectiva da construção

Floyd (1997, 2002) conceitua a *modelagem informática* como um processo de formalização que toma por base *ações situadas* – isto é, ações tomadas em situações concretas, tendo por base informações contextuais (ver Suchman, 2007) – e termina por produzir uma *forma operacional*: uma estrutura de operações inter-relacionadas que descrevem "possíveis maneiras da execução (da realização de operações) em uma área de interesse" (Floyd e Klaeren, 1999, p. 54). Para chegar, a partir de tais descrições, a uma tecnologia executável são necessários artefatos: por um lado artefatos simbólicos como textos ou gráficos, nos quais a forma operacional está documentada, por exemplo, na forma de padrões e procedimentos. Por outro lado, a execução da forma operacional requer artefatos técnicos materiais para sua concretização, como por exemplo, ferramentas e máquinas.

Uma característica específica de Tecnologias da Informação em comparação com outras tecnologias existentes consiste, segundo Floyd, no fato de o computador como artefato técnico viabilizar a execução de artefatos simbólicos na forma de programas de computador: "Dessa forma ambos os aspectos artefato simbólico e técnico são novamente reunidos" (Floyd e Klaeren 1999, p. 71). Tendo o computador como instância de execução, os artefatos simbólicos tornam-se "ativos de maneira praticamente autônoma", justificando, para Floyd (1997), o uso do termo "forma *auto*operacional" nesse contexto.

Com o termo "*modelagem informática*" Floyd compreende o processo completo de produção de software, desde a análise do problema até a programação. Nessa escolha terminológica já se torna visível que o resultado desse processo consiste em um modelo,

o qual representa um determinado domínio (real ou teórico). Nesse ponto, Floyd deixa explícitamente em aberto se a modelagem informática deve ser entendida como *reconstrução* de um domínio ou, na verdade, como uma *construção*, denominando conseqüentemente o processo como "(re)construção operacional" (Floyd e Klaeren 1999, p. 61). A descrição e interpretação de ações situadas oferecem o fundamento para a distinção de operações:

> O observador segmenta o fluxo total dos acontecimentos em pedaços por meio da distinção de operações uma das outras – como [fazê-lo] não é necessariamente dado previamente. Essa descrição serve para possibilitar o entendimento, aprendizado, realização, planejamento ou representação da execução desejada. Dentro desse processo infiltram-se, assim, a perspectiva do observador e seus objetivos (Floyd e Klaeren, 1999, p. 53).

O processo de (re)construção operacional consiste, portanto, na descrição de operações, as quais se exprimem em ações situadas. Como as operações sempre estão embutidas em ações situadas – de modo que ambas não podem ser distinguidas "objetivamente" uma da outra – o processo de (re)construção operacional se trata de um procedimento impregnado pela percepção subjetiva do observador. Além disso, a (re)construção de operações depende do que deve ser modelado, razão pela qual diferentes objetivos possivelmente produzem diferentes (re)construções de operações ou (re)construções de operações distintas.

A diferenciação entre ação e operação, feita por Floyd em referência à teoria da atividade[1], é fundamental para a relação entre ações e tecnologia da informação em sua abordagem: operações consistem em no aspecto das ações passível de descrição formal e com isso de delegação – não apenas de uma pessoa a outra, como também em certos casos a animais adestrados ou artefatos técnicos. Uma característica de operações é portanto sua capacidade de transferência e, por meio desta, as possibilidades de repetição, planejamento e verificação (Floyd e Klaeren, 1999, p. 53). A descrição de operações torna-se *forma operacional* quando diferentes operações são combinadas em um conjunto que descreve sua possível execução em um determinado domínio.

A descrição de operações e sua combinação em uma forma operacional tem por objetivo uma generalização e, com ela, um desprendimento do contexto específico da ação. Essa descontextualização se completa com a criação da forma autooperacional, descrita como uma máquina simbólica (Krämmer, 1988) esquemática e livre de interpretação. Floyd e Klaeren (1999, p. 71) denominam *estrutura operacional* a descrição descontextualizada da forma autooperacional, na qual estão expressas generalizações de situações de uso. Os autores citam como forma de criação de tais generalizações a consideração de categorias de domínios com conhecimentos e procedimentos especiais, tais como: sistemas de saúde (*eHealth*), controle aéreo e bibliotecas. Outra possibilidade para criação de estruturas operacionais é a generalização de procedimentos similares que cruzam as fronteiras de domínios. Floyd e Klaeren citam como exemplos deste caso o diagnóstico, a configuração ou o planejamento – que

[1] Sobre a aplicação da teoria da atividade no contexto de informática ver, por exemplo, Dahme e Raeithel (1997) e Martins (2007).

aparecem em diferentes domínios, mas podem ser generalizados em estruturas fundamentais comuns (Floyd e Klaeren 1999, p. 72). Por fim, pode-se também identificar estruturas operacionais ligadas a padrões típicos de solução de problemas que não têm ligação direta com um domínio específico. Nessa classificação enquadram-se, por exemplo, algoritmos de busca e ordenação, mas também construtos teóricos como árvore binária ou autômato finito.

Para Floyd, portanto, as ações são de alguma forma inscritas em artefatos de tecnologia de informação – ou, mais precisamente, assim o são aqueles aspectos da ação que podem ser descritos e formalizados independentemente de situações concretas de ação. Operações – como padrões de comportamento independentes de situação e em forma de regras – constituem o fundamento para a construção de conjuntos de ações que são descritas em 'máquinas simbólicas'. Esses conjuntos de ações, chamados também de estruturas operacionais, apresentam-se como padrões operativos extra-situativos – isto é, esquemáticos e livres de interpretação –, os quais são interpretados e tornados úteis para a ação em situações concretas. O repertório de operações armazenado nos artefatos constitui também uma área de atuação que restringe as ações concretas: no limite, a ação se torna possível apenas dentro das fronteiras demarcadas pelas alternativas que a tecnologia oferece.

No entender de Floyd, as estruturas operacionais consistem no subconjunto das ações que satisfazem os critérios de possibilidade de formalização e operacionalização. Ainda apenas um subconjunto dessas estruturas é passível de modelagem informática, a saber, o subconjunto que pode ser descrito como procedimento e que satisfaz os critérios de possibilidade de registro por escrito, esquematização e desprendimento de interpretações.

3. A perspectiva do uso

A utilização de tecnologias de informação ou software está diretamente inter-relacionada de um lado às estruturas organizacionais e, de outro, às ações dos atores sociais. Essa "dualidade da tecnologia" foi proposta por Orlikowski (Orlikowski, 1992, p. 401; Orlikowski et. al., 1995, p. 426) como extensão da *Teoria da Estruturação* de Giddens (1984). Por estruturação Giddens (1984) compreende o processo de intermediação prática entre ação e estrutura. Nesse processo, cada ação de atores específicos se apóia numa estrutura que a possibilita, mas ao mesmo tempo dita certas regras e com isso limita as possibilidades dessa ação. Os atores sociais se relacionam em seu agir a estruturas, assim produzindo-as ou reproduzindo-as ao longo do tempo. Dessa forma, a estrutura social existe, mas sua perpetuação depende das ações dos atores; se há uma mudança nas ações pode haver também, em decorrência, uma mudança na estrutura. As estruturas são, portanto, o meio (a mídia) e o resultado da ação: "De acordo com a noção da dualidade da estrutura, as propriedades estruturais de sistemas sociais são ao mesmo tempo o meio e o resultado das práticas que elas organizam recursivamente" (Giddens, 1984, p. 25). Essa inter-relação entre ação e estrutura é, portanto, denominada por Giddens *dualidade da estrutura*.

Ação significa, nesse contexto, a intervenção – meio consciente, meio inconsciente – no mundo social, de forma estabilizante ou desestabilizante. Estruturas são realidades institucionais e duradouras, mas também modificáveis. Elas podem se

apresentar de formas muito variadas, como por exemplo, rotinas, leis, métodos de administração aplicados em organizações, instruções organizacionais, hábitos recebidos e aceitos, equilíbrios de interesses não mais colocados em questão, assim como metáforas, visões e ideais. Estruturas em uma organização constituem, assim, uma realidade operacional, a qual mediante as ações de seus atores é continuamente confirmada ou colocada em xeque – esta última situação implicando em mudança e desenvolvimento.

Organizações, entendidas como sistemas de ação organizada, reproduzem-se mediante a ação de atores competentes em direção a objetivos. Os atores se relacionam em suas interações com um conjunto de regras e recursos – por exemplo, com os procedimentos estabelecidos de registro contábil e de controle – e confirmam ou modificam as estruturas – como, por exemplo, as estruturas hierárquicas organizacionais – ao reconhecê-las ou não em sua ação. Nesse processo, os atores agem de forma reflexiva, isto é, vinculando em suas ações passado, presente e futuro e o comportamento de outros, assim como estruturas. Simultaneamente, agem esses atores recursivamente sobre as estruturas e as perpetuam exatamente por meio dessas ações.

A teoria da estruturação de Giddens complementa, assim, a perspectiva orientada a atores com uma perspectiva orientada a estruturas. Ações passadas cristalizadas em estruturas tornam-se assim visíveis – nelas estão incorporadas também estruturas de poder que não podem ser mudadas arbitrariamente a qualquer momento.

Giddens não faz, porém, nenhuma menção ao papel das tecnologias da informação no jogo das inter-relações entre estrutura e ação. Orlikowski (Orlikowski, 1992; Orlikowski et. al., 1995) estende a teoria de Giddens, propondo as seguintes relações entre os três pólos estrutura, ação e TI/software[2]:

a) Software é criado mediante a ação humana: sistemas de software em organizações surgem por meio da ação criativa de pessoas e são mantidos em funcionamento por medidas como manutenções e adaptações.

b) Atores agem por meio do software, o qual se torna uma mídia para a ação humana: o software possibilita certas ações, ao mesmo tempo em que limita ou proíbe outras. Nisso está também incluída, porém, a opção de agir diferente do modo previsto no software. Apesar disso, o software pode, até certo ponto, obrigar os atores a certas ações.

c) Atores agem sobre as estruturas da organização quando desenvolvem e utilizam software: os atos de desenvolver, usar, mudar ou ignorar o software apoiam-se sempre nas regras e recursos da organização em questão.

d) Por meio das ações dos atores com o software as regras e recursos da organização são confirmados ou modificados: quando os atores usam software (a/b), eles influenciam dessa maneira consciente ou inconscientemente as propriedades estruturais de uma organização.

Sob essa perspectiva, como se dá a relação entre estruturas, ações e artefatos de software? Originalmente Orlikowski propôs uma relação direta entre software e

[2] Para um bom resumo ver (Pape, 2004, pp. 128 ss.).

propriedades estruturais (ver Orlikowski, 2000, pp. 405ss.), ou seja, artefatos de software incorporariam estruturas construídas pelos desenvolvedores, as quais seriam apropriadas no momento do uso pelos usuários. Seguindo essa linha, portanto, estruturas sociais seriam vertidas em software durante o desenvolvimento.

Essa perspectiva se mostra problemática, entretanto, pois ela descreve TI/software como um artefato estático que incorpora estruturas rígidas e fixas, as quais deveriam estar disponíveis para os usuários. Diversos estudos empíricos contradizem tal suposição, mostrando como os usuários freqüentemente improvisam e utilizam as tecnologias de maneiras novas e imprevistas. Além disso, há aí uma inconsistência conceitual com a teoria de Giddens, segundo a qual estruturas são explicitamente não materiais (cf. Giddens, 1984, p. 33; Orlikowski, 2000, p. 406).

Por essa razão, em um trabalho posterior Orlikowski (2000) conceitua a relação entre software e estrutura de maneira diferente: estruturas não são incorporadas no artefato (i.e. imanentes), mas sim *emergentes* a partir da utilização do artefato. Quando pessoas usam um software, elas utilizam as propriedades do artefato, apoiando-se porém ao mesmo tempo em suas próprias capacidades, conhecimentos, estruturas de poder, suposições e expectativas em relação à tecnologia e seu uso. A utilização da tecnologia é estruturada, portanto, a partir das experiências, conhecimentos, atribuição de significados, hábitos, relações de poder, normas e do artefato tecnológico, de forma que os padrões de uso assim emergentes constituem regras e recursos, os quais estruturarão as futuras interações com a tecnologia. Ao longo do tempo constitui-se nesse processo uma estrutura em relação ao uso da tecnologia que Orlikowski denomina “tecnologia na prática” .

4. Desenho integrado de software e organização

Analisando as duas perspectivas apresentadas acima, torna-se claro que tanto na construção como na apropriação/uso de sistemas de software há uma inter-relação entre a organização e o artefato de software. Ambas as perspectivas trazem à luz aspectos relevantes e importantes da influência mútua entre software e organização, porém sua percepção é seletiva. Acreditamos que as duas perspectivas se complementam, uma podendo oferecer um importante esclarecimento sobre os pontos cegos da outra, isto é, sobre os aspectos ignorados ou por ela tratados de forma marginal.

Na perspectiva da construção (Seção 2) os aspectos insatisfatoriamente considerados estão relacionados ao contexto social: o foco no artefato computador e no software como ‘máquina simbólica’, objeto e resultado da modelagem informática, dá pouca ênfase ao significado desse processo para a construção concreta de ações organizacionais, as quais são de crucial importância para a apropriação de artefatos de software. Na perspectiva do uso (Seção 3) vemos a necessidade de considerar mais explicitamente as maneiras projetadas de uso, que têm necessariamente um papel importante na construção do software. Essas projeções incluem as expectativas implícitas sobre a utilização do software que surgem no contexto de desenvolvimento. Estas são raramente refletidas e algumas vezes são articuladas de maneira incompleta, porém sempre influenciam a apropriação de artefatos de software no contexto de uso. Consideramos que essa influência não é satisfatoriamente esclarecida nem na

conceituação original de estruturas imanentes no software de Orlikowski, nem em sua nova compreensão de estruturas emergentes do uso.

Como vimos, o projeto de software está ligado à tentativa de representar rotinas organizacionais em artefatos de software de maneira independente de seus aspectos concretos e situativos. Esse processo pode ser denominado *descontextualização*, um termo que se apóia na percepção e intenção dos desenvolvedores, e que se encontra amplamente na literatura (ver, por exemplo, Halfmann, 1995; Suchman, 2002; Floyd, 1997). A idéia de rotinas organizacionais descontextualizadas e implementadas no software implica, porém, na *recontextualização* das mesmas durante os processos de apropriação e uso de sistemas de software. A seguir, estendemos as duas perspectivas anteriormente analisadas, compreendendo-as dentro de um processo cíclico que contém as fases de *descontextualização* e *recontextualização.* Dessa maneira, buscamos entender o projeto de software como desenho integrado de software e organização, inspirados em trabalhos anteriores na área (ver Rolf 2003; Krause et al., 2006).

4.1 Descontextualização

A fase de *descontextualização* pode ser decomposta analiticamente em passos intermediários, os quais analisamos – seguindo a divisão proposta por Rolf (2003) – divididos nas etapas de *explicitação, algoritmização* e *implementação*.

Na primeira etapa, de *explicitação*, processos de trabalho são observados e descritos e, assim, tornados explícitos num processo de construção e formalização. Diferentemente de Floyd (Seção 2), porém, esse processo de construção em nossa compreensão não apenas inclui a fabricação de um modelo das operações formalizáveis ou operacionalizáveis, mas implica a organização como um todo. Mesmo que, como na descrição de Floyd, a descrições explícitas propriamente ditas se restrinjam aos aspectos formalizáveis de ações, os observadores/desenvolvedores que elaboram tais descrições se baseiam ao menos implicitamente em uma expectativa ou projeção das práticas[3] organizacionais relacionadas, as quais são construídas tacitamente em conjunto com as descrições explícitas. Essas *práticas organizacionais projetadas* têm uma influência decisiva sobre como os processos organizacionais são percebidos e, assim, sobre quais aspectos da realidade organizacional são tidos como relevantes para explicitação.

Na etapa de *algoritmização*, os aspectos operacionalizáveis dos processos organizacionais são abstraídos em uma especificação formal e traduzidos em uma forma passível de implementação em um software. No desenvolvimento orientado a objetos, por exemplo, isso acontece por meio da definição de objetos com suas correspondentes propriedades e métodos. Assim é gerada uma forma operacional (nos termos de Floyd, ver Seção 2), a qual não é completa em si mesma, entretanto, pois toma como premissa determinadas formas de ação com o software por parte dos usuários. As funcionalidades do software e a maneira com que os usuários lidam com essas funcionalidades devem,

---

[3] O termo *práticas* aqui utilizado tem como referência o campo que Reckwitz (2002) denomina "practice theories", no qual o conceito de práticas significa: "um tipo rotinizado de comportamento que consiste de diversos elementos interconectados entre si: formas de atividades corporais, formas de atividades mentais, 'coisas' e seus usos, um conhecimento contextual na forma de entendimentos, saber-fazer [know-how], estados de emoção e conhecimento motivacional." (Reckwitz, 2002, p. 249).

portanto, ser pensadas como uma unidade de ação – fato que se mostra de forma mais clara no projeto de interfaces humano-computador, mas que possui alcance muito maior, chegando até o desenho de processos de trabalho e formas de sociabilidade.

Com a implementação como terceira etapa da descontextualização, um artefato de software é programado para implementar o modelo formal. Nesse ponto, porém, distanciamo-nos do conceito de *forma autooperacional* de Floyd, o qual pressupõe uma transferência neutra e simples "execução" das formas operacionais pelos artefatos de TI. Na prática, a implementação implica muitas vezes em uma negociação com os aspectos realizáveis em uma plataforma e com a compreensão dos programadores acerca das funcionalidades e das práticas organizacionais em torno do software. Assim, esse processo é mais bem descrito como uma tradução – que, como nos mostram os estudos CTS é sempre traição, ver por exemplo Callon (1986) e Latour (2005) –, produzindo compreensões diferentes acerca do artefato software e gerando novas expectativas de práticas organizacionais – o que a experiência de qualquer programador e o dia-a-dia de usuários pode facilmente confirmar.

O produto final desse processo em três etapas pode ser, portanto, separado analiticamente em três partes: (1) a especificação formal, que descreve as características do software e de sua utilização; (2) o artefato material software (que será fundido ao hardware e demais softwares em um artefato TI); (3) as implícitas *práticas organizacionais projetadas*, que se relacionam ao artefato de software. Essa distinção é, no entanto, apenas analítica, pois não se pode compreender um dos elementos (e.g. o artefato), sem implicar simultaneamente os outros (e.g. sua especificação e as práticas organizacionais projetadas).

Durante o desenvolvimento são, então, definidas prescrições para a manipulação do artefato software tanto de maneira explícita (na especificação formal) como implícita (nas práticas projetadas). No entanto, nenhuma afirmação geral pode ser feita nesse ponto sobre até que ponto o software determinará ou como ele influenciará as ações efetivas na organização. As prescrições e expectativas nesse sentido podem assumir diversas formas (ver Coy, 1995), indo desde a completa automação no sentido de delimitar opções de ação (por exemplo, por meio de máscaras de entrada ou campos de múltipla escolha nas telas do software), passando pelo apoio a formas de ação pelo software como ferramenta (por exemplo, na edição de textos), chegando até à utilização aberta dos sistemas de software como mídia para o processamento e transmissão de informações (por exemplo, a Internet e as novas mídias sociais). Via de regra, encontramos na prática formas mistas desses três tipos de utilização.

Decisivo nesse contexto é que ao longo do desenvolvimento de software são construídas *práticas organizacionais projetadas*, ou seja, a *descontextualização* resulta não apenas na produção de um artefato de software, mas também no desenho de um contexto organizacional projetado. Por esse motivo, nossa perspectiva é enxergar o desenvolvimento de software como desenho[4] integrado de software e organização.

[4] A palavra desenho é usada para traduzir a idéia do verbo inglês *design*, que inclui uma dimensão de ação não contida na palavra "projeto" geralmente usada para sua tradução.

4.2 Recontextualização

Na *recontextualização* podemos distinguir analiticamente dois tipos de requisitos: técnicos e organizatórios. Entre os técnicos estão incluídos o hardware necessário para a implantação do software e a correspondente competência técnica para a operação e manutenção do sistema. Para a organização esse tipo de requisito está longe de sempre ser trivial; não obstante, não lhe dedicaremos maior atenção aqui por estar do foco de nossa observação.

Do ponto de vista organizatório os desafios estão relacionados a assegurar uma utilização adequada do software no contexto organizacional. Para tanto, os atores da organização devem ser familiarizados com a forma de uso intencionada para o software e com as alterações necessárias nos seus procedimentos de trabalho. Diversas medidas podem e são comumente adotadas com o intuito de treinar os usuários e divulgar entre eles as práticas necessárias para manipulação do software.

Outra necessidade ainda do ponto de vista organizatório, mas que nem sempre se apresenta de maneira tão clara como a anterior, consiste na adaptação dos processos e da cultura da organização para incorporar o uso do software. A implantação de um sistema de software toca em interesses de diferentes atores em uma organização das mais variadas maneiras. Isso se relaciona, também, aos objetivos com os quais se justifica a introdução de TI/software, muitas vezes baseados em uma expectativa de elevação da eficiência ou melhoria dos processos de trabalho. Os processos de adaptação disparados pela introdução de um novo software podem gerar resistências devido, por exemplo, à frustração de expectativas e até mesmo a contendas micropolíticas sobre esferas de competência. Os conflitos daí resultantes nos permitem suspeitar que se tem dado pouca atenção à historicidade e aos pontos de rigidez formados ao longo do tempo em rotinas organizacionais. A organização parece ser entendida por muitos como uma massa – talvez um tanto tenaz, porém em princípio maleável – que se poderia moldar sem muitos problemas aos requisitos do software.

No entanto, sistemas de software são o resultado da interpretação de processos organizacionais (como vimos anteriormente), interpretação essa que não necessariamente corresponde às necessidades da organização. Esse descompasso pode acontecer (e freqüentemente acontece) no caso de pacotes de software padronizados (*standard software*), os quais tem por base uma concepção muito abstrata do contexto de uso efetivo, e portanto pouco adaptada a este. Esse também pode ser o caso, entretanto, com softwares individuais "feitos sob medida" cujo processo de desenvolvimento tenha se dado alienando os futuros usuários efetivos ou desconsiderando as evoluções e mudanças ocorridas na organização durante o processo.

Para melhor compreender os problemas da recontextualização e suas causas, é necessário voltar à idéia de prescrições que são definidas durante o desenvolvimento do software acerca do uso e do contexto organizacional dos artefatos (Seção 4.1), ou seja, às *práticas organizacionais projetadas*. Essas práticas definem, portanto, um contexto organizacional projetado, o qual nem sempre corresponde à percepção da organização por parte de seus membros, e nem pode ser tido como única possível interpretação "correta" da organização. É verdade que sempre há (pelo menos numa perspectiva otimista) certa correspondência entre os fatos e relações da organização e os artefatos construídos. Entretanto, devido a uma infinidade de fatores – como, por exemplo, a

contingência das decisões de projeto, a relevância de requisitos para o desenvolvimento ou a inevitável seletividade da percepção dos desenvolvedores – não se pode pensar em uma representação sem perdas, ou uma fotografia perfeita da realidade organizacional em um artefato de software.

O contexto organizacional prévio corresponde a um conjunto de *práticas organizacionais pré-existente*s, compartilhadas pelos membros da organização – ainda que não se possa supor, aqui, uma total congruência das diferentes percepções envolvidas. Em contraste, as *práticas organizacionais projetadas* durante a descontextualização implicam uma realidade organizacional diferente da anterior – já de saída diferente pelo fato de incorporar o software em si e, no mais das vezes, também associada a uma organização do trabalho modificada. O fator decisivo aqui é que essa nova forma organizacional não se deixa simplesmente "difundir" ou transmitir sem ruídos (Teixeira e Cukierman, 2007).

Como nos mostram os estudos de Orlikowski analisados na Seção 3, os membros da organização constituem no uso uma realidade organizacional própria, na qual *práticas organizacionais efetivas* são construídas em torno dos artefatos de software. A nova realidade organizacional assim criada muitas vezes diverge daquela sugerida pelas práticas projetadas no desenvolvimento. Na prática, os artefatos são usados de maneira diferente daquela prevista por desenvolvedores, ou mesmo ignorados de forma criativa com o intuito de assegurar, segundo a visão dos atores envolvidos, a capacidade de ação e adaptação frente a novas situações por que passa a organização.

A fase de recontextualização pode ser visualizada, portanto, como um movimento de deriva (ver Ciborra, 2004), isto é, um deslocamento cujas conseqüências não se pode prever completamente: partindo da realidade organizacional e das *práticas organizacionais pré-existentes* há o confronto com as *práticas organizacionais projetadas* durante a descontextualização; esse confronto produz, por sua vez, uma nova organização e suas correspondentes *práticas organizacionais efetivas*. Essas práticas podem, então, servir de base para um novo desenho integrado de um novo software e novas práticas organizacionais, assim desencadeando um processo espiral de evolução organizacional que itera novamente as fases de descontextualização e recontextualização.

5. Considerações finais

Este ensaio toma por base uma análise do processo de desenvolvimento e uso organizacional de software a partir de dois pontos de vista: a perspectiva da construção de Floyd (Seção 2), que enfoca a consideração da formalização de ações sociais em artefatos de software; e a perspectiva do uso de Orlikowski (Seção 3), a qual enfatiza a apropriação social do software e sua integração às estruturas organizacionais existentes. Essas duas perspectivas foram estendidas de forma obter uma compreensão sociotécnica do desenvolvimento de software como desenho integrado de software e organização dentro de um processo cíclico de evolução organizacional que contém as fases de *descontextualização* e *recontextualização.*

Nessa compreensão, despedimo-nos de uma idéia simplista (ainda que bastante difundida) da transposição "neutra" – isto é, direta e sem perdas – de rotinas organizacionais para artefatos de software e, posteriormente, destes para a organização.

Ao invés disso, procuramos dar ênfase ao processo de negociação entre os interesses e as percepções dos atores envolvidos no desenvolvimento e uso de software, levando em conta as diferentes expectativas e projeções a respeito do funcionamento da organização e do software. Com esse intuito, desenvolvemos o conceito de *práticas organizacionais projetadas*, argumentando que o processo de *descontextualização* consiste não somente na produção de um artefato de software, mas também no desenho de um contexto organizacional projetado. Já na fase de *recontextualização*, enfatizamos o confronto das *práticas organizacionais pré-existentes* na organização com as *práticas organizacionais projetadas*, desencadeando, assim, um processo de adaptação e deriva do qual emerge uma nova organização e suas correspondentes *práticas organizacionais efetivas*.

Uma conseqüência dessa compreensão para o desenvolvimento e projeto de sistemas de software consiste na necessidade de considerar com mais propriedade e profundidade o processo de recontextualização. De fato, enquanto que se pode dispor atualmente de um instrumentário abrangente e relativamente maduro para o desenvolvimento de software nas técnicas de engenharia de software, a qualidade do processo de desenvolvimento e dos artefatos a serem produzidos só poderá ser garantida mediante a efetiva integração dos artefatos de software nos processos organizacionais – aos quais ainda não se tem dado, a nosso ver, a devida atenção.

Agradecimentos

Os autores gostariam de expressar sua gratidão ao Prof. Dr. Arno Rolf da Universidade de Hamburgo por seu apoio e incentivo, aos membros da rede Mikropolis pelas discussões que muito contribuíram para o desenvolvimento das idéias contidas neste artigo, e à Fundação Alexander von Humboldt pelo essencial apoio financeiro.

Referências

Callon, M. (1986). Some Elements of a Sociology of Translation: Domestication of the Scallops and the Fishermen of Saint Brieuc Bay. In J. Law (Ed.) Power, Action and Belief: a new Sociology of Knowledge? Sociological Review Monograph. London, Routledge and Kegan Paul. p 196-233

Ciborra, C. (2004). The Labyrinths of Information: Challenging the Wisdom of Systems. Oxford University Press, USA.

Coy, Wolfgang (1995): Automat Werkzeug Medium. In: Informatik Spektrum 18. 1995. 31-38

Dahme, Christian e Raeithel, Arne (1997): Ein tätigkeitsorientierter Ansatz zur Entwicklung brauchbarer Software. In: Informatik-Spektrum 20. 1997. 5-12

Floyd, Christiane (1992). Software Development as Reality Construction, In: C. Floyd, H. Züllighoven, R. Budde, R. Keil-Slawik (Hrsg.): Software Development and Reality Construction. Springer Verlag, Berlin, 1992, p. 86-100

Floyd, Christiane (1993): Steps - a Methodical Approach to Participative Design. In: Communications of the ACM 36(6). 1993. 83-83

Floyd, Christiane (1997): Autooperationale Form und situiertes Handeln. In: Hubig (1997): 237-252

Floyd, Christiane (2002): Developing and Embedding Autooperational Form, In: Dittrich, Y., Floyd, C., Klischewski, R. (eds.): Social Thinking - Software Practice, MIT Press, Cambridge (MA), p. 5-28

Floyd, Christiane/Klären, Herbert (1999): Informatik als Praxis und Wissenschaft. Tübingen: Universität Tübingen

Giddens, Anthony (1984): The Constitution of Society: Outline of a Theory of Structuration. Berkeley: University of California Press.

Halfmann, Jost (1995): Kausale Simplifikationen: Grundlagenprobleme einer Soziologie der Technik. In: Halfmann, Jost, Bechmann, Gotthard, Rammert, Werner (Ed.) (1995): Theoriebausteine der Techniksoziologie. Technik und Gesellschaft, Jahrbuch 8. Frankfurt/Main, New York: Campus. p. 211-226

Hubig, Christoph (Hrsg.) (1997): Cognitio Humana – Dynamik des Wissens und der Werte, Berlin: Akademieverlag

Krämer, Sybille (1988). Symbolische Maschinen. Darmstadt: Wissenschaftliche Buchgesellschaft.

Krause, D., Rolf, A., Christ, M., Simon, E. (2006): Wissen, wie alles zusammenhängt. In: Informatik-Spektrum 29. 263-273

Latour, Bruno (2005): Reassembling the Social. An Introduction to Actor-Network-Theory. Oxford: Oxford University Press

Lippe, Wolfram-Manfred (Hrsg.) (1989): Software-Entwicklung: Konzepte, Erfahrungen, Perspektiven. Berlin: Springer

Martins, Luiz Eduardo Galvão (2007). Activity Theory as a feasible model for requirements elicitation processes. Scientia, 18(1), 33-40.

Orlikowski, Wanda J. (1992). The duality of technology: Rethinking the concept of technology in organizations. Organization Science 3(3). 398–427

Orlikowski, Wanda J./Yates, Joanne/Okamura, Kazuo/Fujimoto, Masayo (1995): Shaping electronic communication. The metastructuring of technology in use. Organization Science 6(4). 423–444

Orlikowski, Wanda J. (2000). Using technology and constituting structures. A practice lens for studying technology in organizations. Organization Science, 11(4). 404-428

Pape, Bernd (2005): Organisation der Softwarenutzung. Theoriebildung und Fallstudien zu Softwareeinführung und Benutzungsbetreuung. Berlin: Logos.

Reckwitz, Andreas (2002): Toward a Theory of Social Practices: A Development in Culturalist Theorizing. European Journal of Social Theory, 5 (2), 243 - 263

Rolf, Arno (2003): Interdisziplinäre Technikforschung und Informatik. Ein Angebot für einen analytischen Orientierungsrahmen. In: Technikfolgenabschätzung Theorie und Praxis, Nr. 3/4, 12. Jg., Nov. 2003

Suchman, L. (2002). Located accountabilities in technology production. Scandinavian Journal of Information Systems, 14 (2), 91-105.

Suchman, L. (2007) Human-Machine Reconfigurations: Plans and Situated Actions 2nd Edition. New York, NY: Cambridge University Press.

Teixeira, C. A. N,. e Cukierman, H. (2007). Why do software process improvement projects fail? Scientia,18 (1), 24-32

Turing, Alan M. (1936): On computable numbers, with an application to the Entscheidungsproblem. Proceedings of the London Mathematical Society, Series 2, Vol.42 (1936 - 37). 230-265