RESEARCH NOTES

Labour Flexibility in the Information Technology Industry: 21st Century Developments

Kommalapati Charitha*

Pursuing Ph.D at the Centre for Informal Sector and Labour Studies, Jawaharlal Nehru University, Delhi.

*Corresponding Author Email : kommalapaticharitha@yahoo.com

Abstract: *This paper deals with the transformational shift from the waterfall model of software development to the agile method, and the restructuring of production processes in the Information Technology Industry. The adaptation of the agile method intensified hegemonic control over value generation and internalisation of capital interests, via contradictory yet coexisting forms of fragmentation of tasks and collaborative labour processes. The current paper situates the blurring of paid and unpaid labour-time, propensity to enter the labour reserve, and internalisation of capital interests to be characteristic of labour in Information Technology, shaped by the agile method. Following this, the labour is classified as new recruits and Individual Contributors (ICs), semi- managerial positions, and (tech and non-tech) managerial positions; incrementally moving from being subjected to the capital's logic towards internalising capital class interests; drawn from the in-depth case studies of 95 IT workers from the Hyderabad Metropolitan Development Authority (HDMA) region. It argues that the progressively increasing integration of managerial tasks into job responsibility, which is central to the agile method, intensifies processes of capital accumulation by reinforcing labour flexibility to suit changing capital requirements.*

Key Words: Information Technology, Agile, Labour, Control,

INTRODUCTION

The 'systems of information' linked to the commercialisation of information (information ownership) are referred to as radical social and technological changes through modifications to property (data/info ownership) regimes (Black and Schiller, 2014); drawn from Websters' accounts of technological innovation, information flows, occupational changes, and expansion of signs and symbols to be characteristic of the Information Society, as it encompassed modifications in the organisation of work beyond mere accumulation of information (Webster, 1995). Fuchs (2014), on the other hand, anchors the emergence of emerging forms of virtual corporations and complicated coordination to be indicative of forms of capital's flexible accumulation; in opposition to the 'fascination of the new' for failing to acknowledge the evolution of socio-economic configurations. The crisis-borne and transforming nature of capitalist production restructures production processes and labour relations, and hence proposes the notion of 'transnational informational capitalism'; to refer to the dialectic between subjectification of information and objectification of the obtained information. The capital's nature of transformational, not radical sublation (Fuchs, 2014) is noted to intensify 'lumpenisation' of labour, and subsequently processes of capital accumulation. Now, the aforementioned literature, however, has not accounted for shifts in systems of production that restructure labour relations. The transformational shift from the waterfall model to the agile method of software development is, hence, acknowledged to be needed to be further examined to make sense of capital's accommodation of 'inconvenient consequences' that intensify surplus value extraction (Patnaik, 1997). The dialectic interaction between technological adaptation and forces and relations of production in the specific context of Information Technology is the problematic, hence, of this paper.

REVIEW OF LITERATURE

The Software Development Life Cycle (SDLC), in the 70s, both globally and in India, adopted various versions of the waterfall model. It linearly progresses from requirement gathering, analysis, design, development, testing, followed by final deployment (the stage where said project goes live), where control over task complexity, scheduling, and budgetary allocation is operationalised through documentation. The rigidity of said documentation of work is acknowledged to be a necessary step, despite slowing down the development process, as it embodies (direct) control over labour processes. The control over labour is noted to be a function of time and technical (and strategic) autonomy and direct control (Barrett, 2004), drawing from Friedman's (1977) notion of direct control and responsible autonomy.However, the waterfall model failed to accommodate project flexibility in terms of changing requirements, which is characteristic of increasing informationalisation of value generation.

The 'limits of planning in a turbulent environment' made way for the formulation of an Agile Manifesto in 2001 that facilitates 'continuous delivery of valuable software' (agilemanifesto.org); central to it are cooperativeness, incrementality and adaptability, where integration of (technical/project) changes take precedence over conforming to a design. It was noted by Swaber and Highsmith, founders of proponents of the agile framework, to be "openly, militantly anti-management in the beginning" in order to eliminate the position of a project manager (Posner, 2022). It is presented as a 'sustainable' alternative due to the 'constant pace' at which software is produced incrementally throughout project completion, focusing on 'self-organising' teams, eliminating needless documentation. Cockburn (2002) noted that the short iterative production cycles, collaborative labour processes, and continuous feedback are the 'heart' of the 'would-be agile' approach (emphasis: original). It is important to note that the shift to the agile method was not because of technological superiority, but an organisation of work that produced continuous (surplus) value. India is stuck in a low-value-added trap by the early 2000s (Parthasarathy, 2004); the shift, hence, to the agile in combination with restrictions to mobility from low to high value-added projects is noted to manufacture a 'privileged precariat' in the case of Indian IT workers (Sardar, 2019).

METHODOLOGY

The labour in the IT was characterised, thusfar, was based on 'work content' into developers, module leaders, project leaders and project managers (Ilavarasan, 2008), and workers' flexibility, virtuality and mobility (Upadhyay and Vasavi, 2008) and 'ambiguity' of organisational and class positions (Narayan, 2023) to constitute professionals, mid management and management. However, said studies assumed the method of software development to be given, by not taking into consideration its dialectical interaction with labour relations. The current paper situates non-separability between labour-time (work) and life, propensity to be pushed into labour reserve, and internalisation of capital interests to be characteristic of labour in Information Technology, shaped by the agile method. The labour is classified as new recruits and Individual Contributors (ICs), semi-managerial positions, and (tech and non-tech) managerial positions; incrementally moving from being subjected to the capital's logic towards internalising capital class interests. In-depth case studies of 95 IT workers are conducted in two phases in the Hyderabad Metropolitan Development Authority (HDMA) region, by relying on convenience methods of data collection due to limitations of access to the list of population and IT workspaces at the time of data collection, due to the post-COVID adaptation of hybrid methods of work (2022-23). 35 of them are new recruits and ICs (inclusive of 2 freelance consultants), 36 hold semi-managerial positions, and 24 hold managerial positions. Table 1, below, showcases the detailed taxonomy of functionalities.

Table 1. Detailed list of tasks performed by IT workers

New Recruits + Individual Contributors	Semi-Managerial Positions	Managerial Positions
Program coding	Determining user and technical requirements	project/module/link management
Debugging	Project specifications, design, and modifications	Documenting project features, functions, and the technical roadmap to materialize them
Testing	System testing and quality assurance	System maintenance and support
System maintenance and support	Software architecture	Recruitments and general management
Determining user requirements	Team/project management	Curate personalised metrics
Import/export data	Internal and external evaluation	External evaluation of project/team members
Documentation	Reminding of deadlines	List of member requirements and satisfaction
	Program coding, debugging, testing, and documentation	Provide group interaction support
	Documentation	Bonus and other forms of compensation
		Facilitate anonymous feedback(s)
		Program coding, debugging, testing, and documentation
		Documentation

RESULTS

The new recruits, junior resources with less than 4 years of work experience, and senior resources who work as Individual Contributors (ICs) distinctly embody a degree of separation between labour-time brought by the capital in exchange for a wage and life; despite varying degrees of control over labour-time and propensity to be fired or shifted onto a bench impromptu. The interviewed newly recruited workers are put in a training program for an "easy transition" towards "handling" work requirements (worker #21); and post-recruitment, they participate in "grooming meetings" that chart out the *sprint* release plan and task assignments as listed by the *lead* or a *senior resource* that they shadow the first 3 to 6 months (worker #19 and worker #21). The *junior resources*, on the other hand, are assigned to a single large project or multiple projects (sub-teams). A majority of themnoted lack of autonomy over task undertakings or in announcing the estimated time it takes to complete a task (worker #10, worker #61, & worker #65); rationalising a work day exceeding 8 hours(worker #10), where the unpaid labour-time ie., time spent in addition to the announced "*bandwidth/velocity"* (Kupiainen et al, 2015) acts as a litmus test to the workers' *commitment* to project requirements (worker-manager #66). However, while task assignments require that they extend their work day, they need not be available to undertake *ad-hoc* tasks, indicating a separation of labour-time with life (worker #52, worker #59 & worker #57); life here is not in reference to the managerial notion of work-life balance, but life outside of the capitalist control over labour-time.

The Individual Contributors (ICs) work as Subject Matter Experts (SMEs) alongside of assigned leads; an IC who is a part of multiple projects noted that he "does not need to know the in and out of all projects" as their technical expertise is recalled as required, disconnecting him from the project at large (worker #2). ICs self-announce task preferences and estimated time(s) (worker #16), and said control over task assignment, estimated task completion time and "flexible work hours" safeguards them from extending their workday, indicating a separation of work and life; additionally, their technical expertise i.e., 'skill' morphs into leverage from being unilaterally pushed into a labour reserve (worker #54). ICs are evaluated on technical certifications (of at least 2 per year), functionality and complexity of the undertaken task, number of escalations and automations that

Table 2. Mapping the average experience and wages (in rupees) of the interviewed IT workers.

	Junior Recruit	IC	Lead	Tech Manager	Manager
Average Yearly Wage	1020000	195000	1740000	344000	215000
Combined Average	1485000				2795000
Average Experience (in years)	2.5	10	8	13	14
Combined Average (in years)	6.5		8		13.5

The positions of a tech lead, senior database administrator, analyst, lead engineer (cloud, data, DevOps, etc), and senior consultant are extended to developers with 3-7 years of experience. They occupy the position of an organisational bridge between managerial positions and the technically skilled workforce, curated to materialise the logic of the capital. Here, Marx's notion of labour expropriation that describes a worker who subjects themselves to exploitation in order to exploit their subordinates, characterises the aforementioned position. The senior resources, here, oversee task completion and train 2-3 junior resources or new recruits (worker #42 and worker #89); oftentimes, these activities go hand-in-hand as task delegation is synced with junior resources' holistic development of the required technical skills by assigning appropriate tasks. The senior resources are noted to be hands-on available (also evaluated on their ability) to handle emergencies, communicate unanticipated delays in task completion so they could be escalated in a time-sensitive manner, generate weekly work reports, and mediate KT sessions (worker #24, worker #73, and worker #62). This sub-category is additionally inclusive of senior resources, simultaneously supporting multiple projects, but without responsibility to oversee their completion. The senior resources and leadsinternalise and reinforce capital interests in the form of extended unpaid labour-time, labour-time availability, work intensity, KPIs, devalued wages, etc,embodying a higher degree of employment security in comparison to other organisational positions. The lead, on the other hand, coordinates a team of 4-10 IT workers by promoting a " socially cohesive" work environment, evaluating their work (performance) and attitude towards work (worker #1 and worker #11). The first-come first-go (FCFG) method of task assignment is deployed by the teamlead in the initial stages of a project whereas "dynamic allotment of tasks" prioritises tasks based on business impact and time sensitivity, and is a prefered method of task allotment for support projects where task criticality takes precedence (worker #18 and worker #24). However, the team leads "shift task priorities after sprint planning" in accordance with their "whims", and assign complex tasks to the experienced and those with bandwidth availability (worker #71); leading to a non-holistic development of skill with a disproportionate brunt work distribution (worker #26). However, the task distribution linked to functionalities (dimensions) of the project and is anchored on the collaborative labour processes (Andrews et al, 2004), contradicting capitals (direct) control over labour; where 'harmonisation' of class interests (Storey in Barrett, 2004) is materialised through a lead, in the agile method.

The agile manifesto charted a 'motivating environment' as a necessity for sustainable software development. However, complexities associated with coordination and conflict resolution among 'self-organising groups' required overlooking by managerial positions (although fewer, in relation to the waterfall method) to maintain a continuous stream of value. The sub-category encapsulates positions of an architect or a program/module manager (tech-centric managerial positions) and administrative positions such as an HR Manager (non-tech-centric managerial positions). The agile methods 'fix' to managers not making sense of the technical aspects of the software development (Greenspun, 2002), ie, remnants of direct control, is the manufacturing of tech manager as an organisational position. The tech managers are paid relatively higher (by approximately 62%) and are relatively shielded (due to the tech-centric 'skill') from being pushed into the labour reserve in opposition to a (non-tech) manager (worker-manager #17); indicating a shift towards hegemonic control by progressively integrating managerial tasks into 'tech' roles as the agile manifesto aimed to discard the role of a manager in toto. The tech managers undertake 'ownership' for end-to-end delivery of project(s) by taking into account differences in regulatory rules for different lines of business. The managers are first-hand respondents to functionality restoration and ad hoc bugs/fixes; for tasks marked as P1, ie, priority of the highest order, the manager/architect oversees task completion and updates the client accordingly. The holistic project requirements are noted to organically shape the "log-in and out time" (worker-manager #24); indicating non-separation of labour-time and life as availability post-work is morphed into a job requirement. They are evaluated on: escalations, "effective handling of (tech-related) incidents", automation(s), project complexity, teams' overall performance, "sense of ownership", and their ability to undertake projects/tasks without requiring additional resources (worker-manager #18). One of the interviewed managers highlighted that they're offered shares in the company, albeit nominal, constituting up to 10 to 30% of basic salary as a "retention strategy" (ibid). The coexistence of the sense of partnership with the increased risk of being replaced is, hence, characteristic of managerial positions. The "ambiguity" between organisational and class positions is alluded to be necessary for capital accumulation by Narayan (2023). This also explains why viewing class as a mix of Wolff's surplus value (generation and appropriation) and Wright's ownership over forces of production is suitable to examine labour relations, specifically in the IT industry.

The facilities and administrative manager, on the other hand, is 'accountable' for the overall administration that is inclusive of monitoring project progress and roadblocks via weekly catch-ups, document work status and worker appraisals, and finance and vendor management; they additionally hold connects (meet) with leads, and set goals every year "according to one's strengths/interests/skillset" in one-on-one meetings with the team members (worker-manager #43). Their lack of technical training, however, is noted by Dyer-Witherford (2015) as obstructing 'harmonisation' of class interests (Storey in Barrett, 2004) as goals (or KPIs) are noted to be set in an unattainable range, due to their lack of familiarity with technical aspects of software development. They also undertake client-related activities such as time sheets (number of hours billed), leave approvals, administrative planning of monthly and annual budgets, monthly payments such as salaries and other standardised expenses, final invoices, quarterly/annual vendor payments, and floor requirements, etc.

The positions of new recruits, junior resources, senior resources, team leads, semi-managerial and managerial positions are characterised by fragmentation of tasks and collaborative labour processes; which upon critical examination, are noted to alienate an IT worker from the collective working class through checkpoints of direct control such as the sprint retrospection and periodic evaluations, enforced through hegemonic control measures inclusive of KPIs, bonuses linked to performance evaluations, and variable pay- which is noted to constitute approximately 10-25% of total wages, and team ratings. The dimensions of capital-labour relations, ie, separability of labour-time and life, labour flexibility, and internalisation of capital control, hence, highlight an integration of managerial tasks as one moves toward (semi) managerial organisational positions. Now, in terms of employment security, although the position of a tech-manager is 'safer' in relation to a (non-tech) manager, instances of senior tech-managers being coerced to voluntarily resign to be replaced by 2 resources (as posted on Naukri.com) for a cheaper wage, indicates a rising propensity to be pushed into the labour reserve in combination with progressively increasing integration of managerial tasks into job responsibilities.

CONCLUSION

The Just-In-Time (JIT) production, in the specific context of software development is referred to as the 'agile method' in reference to the Agile Manifesto. The transformation to agile method by following Burawoy's classification is a combination of a despotic and hegemonic labour regime; where short production cycles (2-4 weeks), periodic evaluations, and variable component of the wage exert direct control over labour processes, and non-separability of labour-time and life and internalisation of capital interests via worker competition and technical efficiency indicate capitals hegemonic control over surplus value extraction reinforced through aspects of direct control.The iterative productive cycles are recognised to exert greater control on labour processes, resulting in the intensification of surplus value extraction. Following Marglin's notion of 'technical efficiency', where modifications to the division of labour do not embody a 'superior' technological approach but reinforce processes of capital accumulation. The dimensions of capital-labour relations are characterised in terms of individualisation of a team player and collaborative labour processes in the face of capital contradictions- standardisation of labour and 'technical efficiency', fragmentation of tasks and socialisation of labour processes, and devaluation of labour and value generation. The subsequent restructuring of capital-labour relations by the agile method is, hence, argued to constitute continuity and intensification of processes of capital accumulation.

REFERENCES:

1. Andrews, Chris, Lair, Craig, & Landry, Bart. (2004). The labour process in software startups: Production on a virtual assembly line? In R. Barrett (Ed.), Management, labour process and software development: Reality bytes. Routledge. https://doi.org/10.4324/9780203502952.

2. Barrett, Rowena. (2004). Managing the software development labour process: Direct control, time and technical autonomy. In R. Barrett (Ed.), Management, labour process and software development: Reality bytes. Routledge. https://doi.org/10.4324/9780203502952.

3. Black, Alistair, & Schiller, Daniel. (2014). Systems of information: The long view. Library Trends, 62(3), 628-662. https://dx.doi.org/10.1353/lib.2014.0009.

4. Cockburn, Alistair. (2002). Agile Software Development Joins the 'Would-Be' Crowd. Cutter IT, 15(1), 6-12.

5. Dyer-Witherford, Nick. (2015). Cyber-proletariat: Global labour in the digital. Pluto Press. https://doi.org/10.2307/j.ctt183p1zg.

6. Friedman, Andrew. (1977). Industry and labour: Class struggle at work and monopoly capitalism. The Macmillan Press. https://doi.org/10.1007/978-1-349-15845-4.

7. Fuchs, Christian. (2014). Digital labour and Karl Marx. Routledge. https://doi.org/10.4324/9781315880075.

8. Greenspun, Philip. (2002). Managing Software Engineers. ArsDigita Systems Journal.https://philip.greenspun.com/ancient-history/managing-software-engineers.

9. Ilavarasan, Vigneswaran. (2008). Software work in India: A labour process view. In C. Upadhya & A. R. Vasavi (Eds.), In an outpost of the global economy: Work and workers in India's information technology industry. Routledge. https://doi.org/10.4324/9780203152003.

10. Kupiainen, Eetu, Mäntylä, Mika, & Itkonen, Juha. (2015). Using metrics in agile and lean software development – A systematic literature review of industrial studies. Information and Software Technology, 62, 143-163. https://doi.org/10.1016/j.infsof.2015.02.005.

11. Narayan, Devika. (2023). Manufacturing managerial compliance: How firms align managers with corporate interest. Work, Employment and Society, 37(6), 1443-1461. https://doi.org/10.1177/09500170221083109

12. Parthasarathy, Balaji. (2004). India's Silicon Valley or Silicon Valley's India? Socially Embedding the Computer Software Industry. International Journal of Urban and Regional Research, 28(3). https://doi.org/10.1111/j.0309-1317.2004.00542.x.

13. Patnaik, Prabhat. (1997). Accumulation and stability under capitalism. Clarendon Press.

14. Posner, Miriam. (2022, March 27). Agile and the long crisis of software. Clouds, 16. https://logicmag.io/clouds/agile-and-the-long-crisis-of-software/.

15. Sardar, Madhumanti. (2019). 'Privileged precariat' of India's software industry [Unpublished doctoral dissertation]. University of Wisconsin-Madison.

16. Upadhya, Carol, & Vasavi, A. R. (2008). Outposts of the global information economy: Work and workers in India's outsourcing industry. In C. Upadhya & A. R. Vasavi (Eds.), In an outpost of the global economy: Work and workers in India's information technology industry. Routledge. https://doi.org/10.1177/097317411000500205.