



UNIVERSIDADE FEDERAL DE SERGIPE
PRÓ-REITORIA DE PÓS-GRADUAÇÃO E PESQUISA
NÚCLEO DE PÓS-GRADUAÇÃO E PESQUISA EM ECONOMIA
MESTRADO PROFISSIONAL EM DESENVOLVIMENTO REGIONAL E GESTÃO DE
EMPREENDIMENTOS LOCAIS

DISSERTAÇÃO DE MESTRADO

OS LIMITES DA SUBSUNÇÃO DO TRABALHO INTELECTUAL NO PROCESSO
PRODUTIVO DE SOFTWARE

JOSÉ GUILHERME DA CUNHA CASTRO FILHO

SÃO CRISTÓVÃO
SERGIPE – BRASIL
MARÇO - 2010

OS LIMITES DA SUBSUNÇÃO DO TRABALHO INTELECTUAL NO PROCESSO PRODUTIVO DE SOFTWARE

JOSÉ GUILHERME DA CUNHA CASTRO FILHO

Dissertação de Mestrado apresentada ao Núcleo de Pós-Graduação e Pesquisa em Economia da Universidade Federal de Sergipe, como parte dos requisitos exigidos para a obtenção do título de Mestre em Desenvolvimento Regional e Gestão de Empreendimentos Locais.

ORIENTADOR: DR. CÉSAR RICARDO SIQUEIRA BOLAÑO

UNIVERSIDADE FEDERAL DE SERGIPE
MESTRADO PROFISSIONAL EM DESENVOLVIMENTO REGIONAL E GESTÃO DE
EMPREENDIMENTOS LOCAIS.
SÃO CRISTÓVÃO – SERGIPE
2010

**FICHA CATALOGRÁFICA
BICEN/UFS**

**FICHA CATALOGRÁFICA ELABORADA
PELA BIBLIOTECA CENTRAL DA
UNIVERSIDADE FEDERAL DE SERGIPE**

OS LIMITES DA SUBSUNÇÃO DO TRABALHO INTELECTUAL NO PROCESSO PRODUTIVO DE *SOFTWARE*

Dissertação de Mestrado defendida por **José Guilherme da Cunha Castro Filho** e aprovada em 29 de Março de 2010 pela banca examinadora constituída pelos doutores:

Dr. CÉSAR RICARDO SIQUEIRA BOLAÑO
Universidade Federal de Sergipe - UFS

Dra. VERLANE ARAGÃO SANTOS
Universidade Federal de Sergipe - UFS

Dr. BENEDITO RODRIGUES DE MORAES NETO
Universidade Estadual Paulista - UNESP

A Lucia Isabel Natividade Oliveira, minha Mãe, por ser para mim um grande exemplo de dedicação, caráter e bondade.

A Aline dos Reis Souto Castro, minha esposa, e Felipe Souto Castro meu filho por serem sempre as minhas amáveis e inseparáveis companhias.

Ao meu orientador César Ricardo Siqueira Bolaño, que acreditou no projeto desde a primeira idéia me incentivando com toda a paciência e dedicação. Sem a sua liderança nada disso seria possível.

AGRADECIMENTOS

Aos amigos do mestrado Helom Oliveira e Luciane Azevedo que me incentivaram durante todos os altos e baixos dessa jornada.

Ao grande amigo **Robsom Marins**, sempre presente nas minhas mais importantes conquistas.

À **Professora Verlane Aragão Santos** pelas contribuições iniciais que puderam abrir e clarear melhor todo o caminho.

Ao meu **Pai José Guilherme Castro**, que pôde me mostrar rotas alternativas durante o percurso da vida.

Aos meus **Avós Maternos Thereza e Manoel Dias (*In memoriam*) e Paternos Celina e Geraldo Castro (*In memoriam*)** por serem eternas fontes de inspiração em tudo aquilo que faço.

A toda a **minha família e a família da minha esposa** por compreenderem a distância e a dedicação que tenho ao trabalho.

Todo indivíduo é o proprietário de uma porção da força de trabalho da comunidade, da sociedade e da espécie. Esta consideração constitui o ponto de partida para a teoria do valor do trabalho, que os economistas burgueses acham poder seguramente desprezar, porque estão interessados não nas relações sociais mas nas relações dos preços; não no trabalho mas na produção, não no ponto de vista humano mas no ponto de vista burguês.

Harry Braverman (1920-1976)

RESUMO

O princípio da produção moderna é combinar novas técnicas de conhecimento, aliadas a processos de desenvolvimento computadorizados cada vez mais sofisticados para aumento da produção e aperfeiçoamento contínuo do próprio modelo. A Terceira Revolução Industrial, caracterizada pela extrema especialização do trabalho intelectual e a tendência de apagamento das fronteiras entre trabalho manual e intelectual, deve ser analisada sob o prisma da automação com base microeletrônica, suas implicações históricas e conseqüências ao longo do processo produtivo. O objetivo deste texto é contribuir para essa discussão ao estudar a codificação do trabalho intelectual a partir da criação de artefatos computacionais, ou mais especificamente, do estabelecimento das linguagens e métodos computacionais que permitem o amplo processo de codificação do conhecimento, fruto do desenvolvimento e produção de *software* com esta finalidade. O texto efetua a análise de dois tipos distintos de subsunção do trabalho intelectual. O primeiro tipo, ligado ao surgimento da linguagem de programação, é caracterizado pelo enquadramento do programador às ferramentas de *software* voltadas à programação. Essa atividade, fortemente atrelada às linguagens de programação e metodologias de desenvolvimento de *software*, é requerida para a concepção do programa de computador em um processo de criação de *softwares* por meio de *softwares*. Nesse tipo de produção, um *software* assume o papel de ferramenta de concepção e desenvolvimento de um produto que é também um *software*. Em outros termos, os sistemas CAD/CAM são construídos a partir de ferramentas de *software* cujo processo de desenvolvimento é marcado por fases similares às da produção industrial automatizada: concepção e execução. O segundo tipo de subsunção é decorrente do produto originado do primeiro tipo. Os *softwares* desenvolvidos para a indústria, tal como o CAD/CAM, permitem tanto a concepção planejada dos produtos industriais quanto a execução objetivada operada pelas máquinas automatizadas controladas por *softwares*. Os *softwares* industriais desenvolvidos para serem executados nas máquinas automatizadas são passíveis de reprogramação no chão de fábrica através de interfaces simplificadas nos Controles Numéricos Computadorizados – CNC, expandindo dessa forma as atividades dos operadores das máquinas e conseqüentemente ampliando os limites de subsunção de trabalho no capital.

Palavras-chaves: Trabalho. Subsunção do trabalho intelectual. Terceira Revolução Industrial.

ABSTRACT

The principle of modern production is to combine modern production techniques, new knowledge, combined with increasingly sophisticated computerized development processes to increase production and processing continuous model itself. The Third Industrial Revolution, characterized by extreme specialization of intellectual labor and tendency to erase the borders between manual and rights, must be analyzed from the perspective of automation based microelectronics, its historical implications and consequences over production process. This paper aims to contribute to this discussion studying the codification of intellectual work from the creation of computing artifacts, or more specifically, the establishment of languages and computational methods that allow the broad process codification of knowledge, fruit of development and production of software for this purpose. The text makes the analysis of two separated types of intellectual work's subsumption. The first type, connected the emergence of the programming language, is characterized by the programmer framework to software tools aimed at the programming process. This activity is strongly linked to language programming methodologies and software development, is required to design the computer program in a process of creating software by software. In this type of production, the software assumes the role of design tool and develops a product that is also a software. In other words, the CAD / CAM systems are built from software tools within a development process marked by similar stages as the automated industrial production: design and implementation. The second type of subsumption is due to the product originated from the first type. Software developed for industry, such as CAD / CAM, allow both the planned design of industrial products and the objectified execution operated by automated machines controlled by software. The industrial software designed to run on machines automated are capable of reprogramming on factory floor through simplified interface in Computer Numerical Control - CNC, thereby expanding the activities of machine's operators and thus extending the boundaries of work in subsumption capital.

Key Words: Work. Subsumption of intellectual work. Third Industrial Revolution.

SUMÁRIO

Introdução	1
Capítulo 1	4
Uma história da máquina-ferramenta e a subsunção do trabalho	4
Capítulo 2	40
Linguagens e ferramentas computacionais.....	40
Item 2.1	46
Histórico resumido do desenvolvimento do <i>hardware</i> e do <i>software</i>	46
Item 2.2	68
As linguagens de <i>software</i>	68
Capítulo 3	77
Produção de <i>softwares</i> e a subsunção do trabalho intelectual	77
Análises conclusivas.....	114
Bibliografia.....	119
Anexos.....	124
I - Evolução das linguagens de programação	124
II - Glossário	125

Abreviaturas e Siglas

CAD	Computer-Aided Design (CAD) ou desenho assistido por computador
CAE	Computer Aided Engineer ou engenharia assistida por computador
CAM	Computer Aided Manufacturing ou manufatura assistida por computador
CASE	Computer-Aided Software Engineering ou engenharia de software assistida por computador
CMMI	Capability Maturity Model Integration ou modelo de capacitação de modelagem integrada
CN	Controle Numérico
CNC	Controle Numérico Computadorizado
CPU	Central Processing Unit ou unidade de processamento central
CSN	Companhia Siderúrgica Nacional
DNC	Direct Numerical Control ou controle numérico direto
EDVAC	Electronic Discrete Variable Automatic Computer ou computador eletrônico de variável discreta
ENIAC	Electrical Numerical Integrator and Computer ou computador elétrico de integração numérica
EUA	Estados Unidos da América
FDD	Feature-Driven Developmen ou desenvolvimento dirigido à característica
FMS	Flexible Manufacturing System ou sistema de manufatura flexível
IBM	International Business Machines ou máquinas de negócio internacional
ICG	Interactive Computer Grafics ou gráficos de computador interativos
ISO/IEC	International Organization for Standardization/International Electrotechnical Commission ou comissão internacional para padronização eletrotécnica
ITA	Instituto Tecnológico de Aeronáutica
LCD	Liquid Crystal Display ou visor de cristal líquido
MFCN	Máquinas-Ferramenta com Controle Numérico
MFCNC	Máquinas-Ferramenta com Controle Numérico Computadorizado
MFU	Máquina ferramenta Universal
RUP	Rational Unified Process ou processo racional unificado
SEI	Software Engineering Institute ou instituto de engenharia de software
SFM	Sistemas Flexíveis de Manufatura
TIC	Tecnologias da Informação e Comunicação
TICC	Trabalho Intellectual de Concepção Criativa
TIERPP	Trabalho Intellectual de Execução de Rotinas Pré-Programáveis
UNIVAC	Universal Automatic Computer ou computador automático universal
XP	eXtreme Programming ou programação extrema

LISTA DE TABELAS

Tabela 1 – Evolução dos sistemas CAD conforme a evolução dos computadores e linguagens de programação (FELDENS, 2000, p. 8) e anexo I, evolução das linguagens de computação (LÉVÉNEZ, 2007)	79
--	----

LISTA DE FIGURAS

Figura 1 – Representação da evolução do processo de trabalho.	22
Figura 2 – Aplicações de produtos de <i>software</i> na indústria.	44
Figura 3 – Processo de cálculo da máquina diferencial, baseado em PIROPO (2006).	48
Figura 4 – Modelo de entrada, processamento e saída de dados da máquina analítica de Babbage baseado em MARTIN, M.B & LOCH, M. (1999).	50
Figura 5 – Estruturas básicas para a programação baseado em GONICK (1986).	52
Figura 6 – Arquitetura básica de <i>hardware</i> e <i>software</i> (Desenvolvido com base em PACITTI, 1988).	69
Figura 7 – Exemplo de instruções em linguagem de alto nível e estruturada.	72
Figura 8 – Instruções para o somatório de duas variáveis em diferentes níveis de linguagem.	72
Figura 9 – Seqüência de instruções para o somatório de duas variáveis em linguagem de máquina.	73
Figura 10 – Seqüência de instruções para o somatório de duas variáveis em linguagem de baixo nível.	73
Figura 11 – Abstração do conhecimento do hardware possibilitado pela linguagem de alto nível.	74
Figura 12 – Conversão da linguagem de alto nível para a linguagem de máquina feita pelo compilador.	75
Figura 13 – Ciclo de projeto de produto (FELDENS, 2000, p. 6).	81
Figura 14 - CAD/CAM: Sistema Integrado de Produção. UFMG (DEMEC/EE/UFMG - http://www.demec.ufmg.br/Grupos/Usinagem/CADCAM.htm)	85
Figura 15 – Processo de produção com ferramentas cad/cam (CASTELLTORT, 1988, p. 42 e 43).	86
Figura 16 – Ciclo de vida clássico de software (PRESSMAN, 1995).	96
Figura 17 – processo de desenvolvimento de <i>software</i> (ferramentas) CAD-CAM para utilização em máquinas automatizadas.	103
Figura 18 – Escopo de atuação da fábrica de software (FERNANDES, 2007).	105
Figura 19 – Escopo de atuação da fábrica de <i>software</i> conforme a complexidade dos processos de produção (FERNANDES, 2007).	106

INTRODUÇÃO

Os sistemas de inovação ligados à dinâmica da produção capitalista hoje combinam o desenvolvimento de técnicas e conhecimentos específicos aos diferentes setores da economia, com o de procedimentos e linguagens informacionais cada vez mais sofisticados, as quais possuem um caráter geral, atuando sobre um conjunto muito grande de setores produtivos. A Terceira Revolução Industrial deve ser analisada não simplesmente sob o prisma da automação com base microeletrônica, mas essencialmente como processo de subsunção do trabalho intelectual, caracterizado por uma profunda especialização deste e por uma tendência de apagamento das suas fronteiras em relação ao trabalho manual. As suas implicações históricas e suas conseqüências para os processos produtivos se traduzem no desenvolvimento de um novo paradigma industrial. Tal processo se caracteriza mais especificamente pelo estabelecimento das linguagens e métodos computacionais que garantem a existência do amplo modelo de codificação do conhecimento, permitido pelo desenvolvimento e produção de *softwares* com essa finalidade.

O presente trabalho tem como objetivo verificar os limites da subsunção do trabalho intelectual no processo produtivo industrial, a partir da análise das ferramentas computacionais de concepção (*design*) e manufatura, CAD e CAM¹ e sua simbiose com as máquinas-ferramenta automáticas. Além da forma acabada do *software*, que permite a concepção e manufatura na produção industrial, são analisadas as ferramentas capazes de conceber o próprio *software*, no caso as ferramentas CASE, em um processo de produção de *software* por intermédio de *software*. O objeto empírico é a história da linguagem

¹ CAD/CAM, são sistemas de *software* que auxiliam, respectivamente, a concepção (*design*) e criação (execução) de peças piloto (que uma vez definidas no computador por intermédio do *software* serão produzidas continuamente de forma automatizada nas máquinas CNC) através de uma interface computadorizada.

computacional, entendida como ferramenta nos processos de trabalho contemporâneos, sob dois pontos de vista. Primeiro, como ferramenta, no sentido de instrumento inacabado e em processo de aperfeiçoamento que é fornecido ao trabalhador para que este desempenhe as suas atividades industriais e, segundo, como instrumento de trabalho propriamente dito, inserido como parte integrante do processo produtivo de *software*, no qual assume o papel de pré-determinar as ações do trabalhador, codificando o seu conhecimento em um processo crescente de transformação e subsunção do trabalho intelectual. A idéia da subsunção do trabalho intelectual no capital, permitida justamente pelo desenvolvimento das tecnologias informacionais, paralela ao da intelectualização geral de todos os processos de trabalho, pode ser tomada como uma chave interpretativa para definir o conceito de Terceira Revolução Industrial (BOLAÑO, 1995).

No primeiro capítulo, histórico, são analisadas as mudanças no trabalho à medida que são introduzidas, primeiro, as máquinas-ferramenta que, com a Revolução Industrial acabam por adquirir papel fundamental no processo produtivo, tornando o trabalho redundante e subordinado-o realmente ao capital. Num segundo momento, de acordo com Marx (1980, p. 441), “a indústria moderna teve então de apoderar-se de seu instrumento característico de produção, a própria máquina, e de produzir máquinas com máquinas” em um processo ocorrido durante a Segunda Revolução Industrial. Essa tendência assume um estágio superior, completando o sistema das máquinas que posteriormente passarão a ser programadas por *softwares* de computador. Bolaño (1995) aponta a importância das ferramentas computacionais na Terceira Revolução Industrial, como um processo de avanço da subsunção do trabalho intelectual, seguindo uma linha de raciocínio similar à de Marx na sua análise da passagem da manufatura à grande indústria, isto é, da subsunção formal do trabalho à sua forma real, ocorrida durante a Primeira Revolução Industrial.

O objetivo deste trabalho é justamente avançar nessa comparação, estudando inicialmente, no segundo capítulo, a história do computador e das linguagens computacionais que, na medida em que se desenvolveram, possibilitaram o pleno domínio das máquinas e, com isso, um maior controle do processo de trabalho. Em seguida, no terceiro capítulo é discutida a simbiose entre as linguagens computacionais e as máquinas-ferramenta, de forma a apresentar o processo de produção de *software* a partir das linguagens computacionais como um processo inacabado de subsunção, similar ao ocorrido durante a passagem do trabalho manufatureiro ao industrial durante a Primeira Revolução Industrial.

Tal processo inacabado consiste, por exemplo, na produção e desenvolvimento de *softwares* com a finalidade de controle de máquinas industriais, no sentido dos programas de computadores controlarem todo o processo produtivo. O desenvolvimento de tais *softwares* é abordado enfatizando a importância vital do trabalhador durante a fase de concepção dos mesmos e dos mecanismos utilizados durante a produção para reduzir tal importância. Nesse sentido é abordado o limite da subsunção na produção de *software* por meio de *software* que resulta em um produto final acabado que adquire, em um segundo momento, papel fundamental de dominação e controle do processo de trabalho no chão de fábrica, acentuando dessa forma o processo de exclusão do trabalhador desencadeado desde a primeira Revolução Industrial. O processo de produção de *software* por meio de *software* é extremamente peculiar no sentido em que a produção é conduzida através de ferramentas de desenvolvimento de *software* em sua constante interação e aperfeiçoamento em conjunto com o trabalhador (desenvolvedor ou programador) em um trabalho extremamente intelectualizado. Nesse processo produtivo, esse trabalhador, altamente qualificado, precisa adequar-se à forma de concepção do *software* para a sua produção.

CAPÍTULO 1

UMA HISTÓRIA DA MÁQUINA-FERRAMENTA E A SUBSUNÇÃO DO TRABALHO

As bases da cooperação fundada na divisão do trabalho, que moldaram os inícios da produção capitalista operam a partir da organização do trabalho manufatureiro, em oficinas organizadas sob comando capitalista, de indivíduos com diferentes habilidades e destrezas, com o objetivo de produzir produtos específicos. Conforme Marx (1980, p. 393),

a manufatura se origina e se forma, a partir do artesanato, de duas maneiras. De um lado, surge da combinação de ofícios independentes diversos que perdem a sua independência e se tornam tão especializados que passam a constituir apenas operações parciais do processo de produção de uma única mercadoria. De outro, tem sua origem na cooperação de artífices de determinado ofício, decompondo o ofício em suas diferentes operações particulares, isolando-as e individualizando-as para tornar cada uma delas função exclusiva de um trabalhador especial. A manufatura, portanto, ora introduz a divisão do trabalho num processo de produção ou a aperfeiçoa, ora combina ofícios anteriormente distintos. Qualquer que seja, entretanto, seu ponto de partida, o resultado final é o mesmo: um mecanismo de produção cujos órgãos são seres humanos.

Durante a Primeira Revolução Industrial, ocorre o processo histórico da passagem da subsunção formal à subsunção real do trabalho no capital (MARX, 1980), em que o conhecimento acumulado em indivíduos com habilidade de desenvolvimento de produtos específicos - através de um processo que recolhe a herança do artesanato medieval que produziu aquele conceito - se submete à ação do trabalho incorporado no capital. O crescente desmembramento das atividades individuais em pequenas tarefas objetiva a redução do tempo despendido na produção, assim como do esforço envolvido. O conhecimento encerrado no indivíduo, de acordo com o modo de produção artesanal, é o resultado do acúmulo de conhecimento repassado através das gerações de acordo com as regras da relação mestre-aprendiz. Isso permite ao indivíduo artesão total controle do processo produtivo. Com a divisão de trabalho que emerge a partir da manufatura, esse

conhecimento tende de alguma forma a ser compartilhado entre os envolvidos no processo de trabalho, que adquire a forma geral de trabalho cooperativo. Em todo caso, as operações propriamente intelectuais e o controle do processo no seu conjunto vão sendo crescentemente concentrados no capital. A esse processo característico da manufatura, descrito por Marx (1980), Bolaño chama de “acumulação primitiva do conhecimento” (BOLAÑO, 2000, p. 46, 47).

Conforme Marx, no capítulo décimo do livro primeiro d’o Capital (MARX, 1980), a manufatura se organiza sob duas formas distintas, que podem se apresentar combinadas entre si. Essa duplicidade de formas de organização, que será herdada pela grande indústria baseada na maquinaria, como se verá adiante, está diretamente ligada às características do produto, que pode ser produzido parcialmente, de forma independente, por diferentes grupos de indivíduos até finalmente ser montado, constituindo um simples ajustamento mecânico de produtos parciais independentes ou de forma seqüencial, em operações e manipulações conectadas até que finalmente se apresente em seu estágio final de produto acabado. No primeiro caso, caracterizado por Marx como **manufatura heterogênea** (MARX, 1980), as operações parciais podem ser executadas como ofícios independentes entre si, tal como na manufatura de um relógio, em que trabalhadores independentes e em locais distintos produzem as diversas peças individuais para que seja então montado e entregue funcionando como produto manufaturado. A manufatura de tais produtos se dá com a cooperação direta dos trabalhadores parciais sob o comando do mesmo capital no fracionamento da produção em processos **heterogêneos** e pouco permite o emprego de instrumentos e ferramentas comuns de trabalho, sendo menos vantajosa para o capitalista, devido à necessidade de construção e manutenção de um sistema relativamente disperso de produção. A falta de conexão entre os processos em que

se decompõe a produção de objetos produzidos de forma independente entre si dificulta a manufatura de produtos de tal natureza em uma indústria mecanizada.

No segundo tipo de manufatura, chamada por Marx **manufatura orgânica** (MARX, 1980, p. 398), os artigos produzidos percorrem fases de produção conexas, de forma gradativa, tal como na produção de agulhas, em que o arame passa pelas mãos de 72 a 92 trabalhadores parciais, realizando cada um uma tarefa específica no processo produtivo. Conforme Adam Smith (1983, p. 43),

esse grande aumento da quantidade de trabalho que, em consequência da divisão do trabalho, o mesmo número de pessoas é capaz de realizar, é devido a três circunstâncias distintas: em primeiro lugar, devido à maior destreza existente em cada trabalhador; em segundo, à poupança daquele tempo que, geralmente, seria costume perder ao passar de um tipo de trabalho para outro; finalmente, à invenção de um grande número de máquinas que facilitam e abreviam o trabalho, possibilitando a uma única pessoa fazer o trabalho que, de outra forma, teria que ser feito por muitos.

Nesse tipo de manufatura, a partir do momento em que a matéria-prima adentra o processo produtivo para ser transformada em produto, o resultado do trabalho de cada trabalhador parcial resulta em insumo para o trabalhador imediatamente conectado a este. Trata-se, de duas formas fundamentais que desempenham papéis inteiramente distintos na transformação da grande indústria baseada na maquinaria.

A divisão das atividades em tarefas menores, empregada na manufatura, entre o século XVI e a segunda metade do século XVIII, tem o intuito de diminuir o tempo de desenvolvimento e execução da atividade, assim como o esforço envolvido. As tarefas isoladas são repetidamente melhoradas a fim de se constituírem conjuntamente em um

processo de trabalho realizado no menor tempo de execução e com o menor esforço empregado. A divisão de tarefas em partes menores permite ao trabalhador um melhor desempenho na execução, aumentando a destreza pela repetição, às custas do conhecimento total de cada fase do processo de trabalho e, portanto, de um atrofiamento de determinadas funções físicas e intelectuais em benefício de uma única. Nesse sentido, ao invés de um único artesão necessário para produzir um determinado produto, tem-se grupos de trabalhadores desenvolvendo produtos de forma coordenada, a serviço do capital produtivo.

O período manufatureiro estabelece como princípio, através da sistemática divisão do trabalho em diferentes fases de produção e trabalhadores coletivos parciais, a diminuição do tempo de trabalho para a produção de mercadorias. Nas diferentes fases de produção e na maioria dos ofícios envolvidos, a manufatura cria uma classe de trabalhadores com baixa ou nenhuma destreza especial. O desenvolvimento de uma única especialidade limitada, em detrimento da capacidade total de trabalho do indivíduo, do qual se exige pouca ou nenhuma formação, comparando-se com o artesanato, é explicado assim por Marx (1980, p. 403, 404):

As diferentes operações executadas sucessivamente pelo produtor de uma mercadoria e que se entrelaçam no conjunto de seu processo de trabalho apresentam-lhe exigências diversas. Numa tem de desenvolver mais força; noutra, mais destreza; numa terceira, atenção mais concentrada etc.; e o mesmo indivíduo não possui, no mesmo grau, essas qualidades. [...] O hábito de exercer uma função única limitada transforma-o naturalmente em órgão infalível dessa função, compelindo-o à conexão com o mecanismo global a operar com a regularidade de uma peça de máquina.

A destreza otimizada na execução repetida das tarefas fracionadas se estabelece como fator motivador para o capital no aperfeiçoamento contínuo desse modo de produção. A manufatura tem como pilar central, a força de trabalho, elemento a partir do

qual se desdobra o processo que cria e amadurece as pré-condições que permitem o advento da Primeira Revolução Industrial. Tal revolução foi possível devido à acumulação do conhecimento extraído dos contingentes de trabalhadores, organizados em ofícios, detentores de habilidades manuais que, ao serem agrupados em grandes contingentes de maneira coordenada e articulada, constituíram mecanismos produtivos vivos, com partes formadas essencialmente de seres humanos. A especialização contínua e gradual de cada parte do processo produtivo permitiu tanto a decomposição de tarefas quanto a especialização de uma parcela crescente de trabalhadores, cujas tarefas crescentemente repetitivas, tornaram-se susceptíveis de serem automatizadas. A qualificação do contingente crescente de trabalhadores técnicos especializados criado a partir da Primeira Revolução Industrial é significativamente distinta daquela que constitui inicialmente a base técnica artesanal. Há, nessa passagem, como mostra Marx (1980, p. 394), uma desqualificação e rebaixamento da força de trabalho empregada no processo produtivo, de modo que, qualificação e especialização não devem ser tomadas como sinônimos, mas como movimentos tendenciais essencialmente antagônicos de evolução das características objetivas da força de trabalho no capitalismo produtivo.

Descendo ao pormenor, vê-se, de início, que um trabalhador que, durante sua vida inteira, executa uma única operação transforma todo o seu corpo em órgão automático especializado dessa operação. Por isso, levará menos tempo em realizá-la que o artesão [por exemplo] que executa toda uma série de diferentes operações. O trabalhador coletivo que constitui o mecanismo vivo da manufatura consiste apenas nesses trabalhadores parciais, limitados. Por isso, produz-se em menos tempo ou eleva-se a força produtiva de trabalho, em comparação com os ofícios independentes.

No capítulo sobre maquinaria e grande indústria, Marx enfatiza a importância do surgimento da máquina-ferramenta no desenvolvimento da Primeira Revolução Industrial, em que aquela, mais do que um simples mecanismo de trabalho, representa a transferência sistemática do conhecimento e habilidades do trabalhador sobre o processo de trabalho

para o capital constante. O ritmo das atividades ditado pela máquina acentua a dependência do trabalhador no processo produtivo. As máquinas-ferramenta, mecanismos móveis que permitem a elaboração de produtos a partir da transformação de material em bruto, foram historicamente elaboradas a partir da observação de cada atividade individualizada e dividida do trabalho na manufatura. Os *engineers*² elaboradores de tais máquinas, a partir de sistemáticas observações de movimentos repetitivos, materializaram o conhecimento do trabalhador nesses objetos, que gradualmente excluíram o próprio trabalhador do processo de trabalho, tornando-o coadjuvante onde inicialmente assumira papel principal por deter as virtuosidades necessárias para o desempenho das atividades.

No modelo capitalista de produção de mercadorias, o processo de valorização é dominante em relação ao processo de trabalho, que se apresenta como subordinado a este. Desse modo, a organização manufatureira do trabalho se apresenta como um problema para o capital devido à subordinação do processo de trabalho à habilidade do trabalhador. Esse modo de produção apresenta limitações incompatíveis com a necessidade de produção criada pelo capital na medida em que a base técnica (artesanal) dificulta, em um primeiro momento, o processo de decomposição do trabalho devido a um único trabalhador congrega as habilidades necessárias à produção de uma determinada mercadoria em todas as suas fases de processamento. Devido à dificuldade inicial de divisão do trabalho ocasionada pela abrangente dominância do ofício dos trabalhadores, a manufatura se divide hierarquicamente entre trabalhadores especializados (oficiais) e trabalhadores não-especializados (peões).

² Taule usa o nome de *engineers* que, diferentemente dos engenheiros contemporâneos, eram trabalhadores manuais extremamente qualificados, com grande herança da base técnica artesanal, e que construíam, eles próprios, mecanismos e máquinas rudimentares – *engines* (TAUILE, 2001).

Essa divisão hierárquica implica que parte da força de trabalho está, de fato, fora do controle do capital, tanto no processo produtivo quanto no de aprendizagem, ambos vitais para a dominância do processo de valorização em relação ao processo de trabalho. A manufatura significa, portanto, para o capital, a sua dependência em relação à força, destreza e virtuosidade do trabalhador especializado e suas ferramentas individuais. Os trabalhadores especializados formam uma categoria de trabalhadores reduzida, da qual depende a velocidade do desenvolvimento e os avanços nas modalidades de produção.

Nesse sentido, a manufatura se constitui como um processo limitante, sob o ponto de vista do capital, devido tanto à relativamente baixa velocidade em que o trabalho pode ser desempenhado, conforme as suas funções, quanto à dinâmica de inovação. As máquinas produzidas sob o sistema manufatureiro vão paulatinamente subordinando o conhecimento especializado, de maneira mais contundente, a partir do desdobramento da segunda forma de manufatura (orgânica), que, antes de limitar-se a aproveitar as condições para a cooperação tal como esta se apresenta, cria as pré-condições a partir da decomposição das atividades do artesão, conforme o aprisionamento de cada trabalhador a uma única função de ofício. O caráter cooperativo em sua forma mais simples é assumido no segundo tipo de manufatura conforme o seu formato orgânico, na medida em que diferentes tipos de trabalhadores estão agrupados conforme tal divisão manufatureira do trabalho, que permite, além da divisão qualitativa na forma de diferentes ofícios, a criação de uma relação matemática fixa quanto ao número relativo de trabalhadores necessários à execução das tarefas nas diversas fases de produção (TAUJLE, 2001).

O aperfeiçoamento ocorrido a partir da relação contínua entre os trabalhadores qualificados e os não-especializados e suas respectivas ferramentas (posteriormente máquinas-ferramenta), acentua-se conforme o desenvolvimento histórico da tecnologia e seu desdobramento nas relações de trabalho. A partir de tais relações é possível classificar o sistema de máquinas em 3 grupos. O primeiro grupo, das máquinas que constituíram as primeiras formas de automação industrial, se caracteriza pelo desenvolvimento de máquinas cuja operação se apresenta como uma extensão do próprio mecanismo em funcionamento, pois, conforme Marx (1980, p. 441) “esse dispositivo mecânico não substitui uma ferramenta qualquer, mas a própria mão humana, que cria uma forma determinada no material de trabalho, o ferro por exemplo, utilizando o gume de instrumentos cortantes etc”. Tais máquinas são exemplo de automação rígida, ou seja, as operações executadas estão diretamente ligadas às funções específicas para as quais foram construídas, não permitindo conseqüentemente a execução de diferentes operações ao longo do processo de trabalho. A automação descrita pelos clássicos no século XIX, é rígida, ou dedicada. Esse tipo de automação está relacionado à superação da dependência do processo produtivo frente à habilidade dos trabalhadores. Conceitualmente, a automação clássica, com base eletromecânica, é estritamente rígida, não existindo, portanto meios que permitam ao sistema de máquinas a alteração dos movimentos para a fabricação de diferentes produtos, conforme as necessidades da produção. Para que esse tipo de automação fosse viável economicamente, era necessário haver uma escala de produção de um único produto, elevado o suficiente para que valesse a pena contratar o minucioso trabalho dos trabalhadores manuais qualificados que as produziam (TAUILE, 2001). Tais máquinas eram manuseadas por trabalhadores altamente qualificados e especializados, que poderiam ser classificados como operadores de equipamentos devido ao grau de especialização envolvido na atividade. No período manufatureiro, ainda sem o

advento do aço, tais máquinas-ferramenta eram manufaturadas integralmente em madeira, o que aumentava o desgaste das peças e conseqüentemente as intervenções para manutenção e ajustes por parte de trabalhadores altamente qualificados. As constantes manutenções, aliadas ao fato de que as máquinas-ferramenta eram encomendadas para o desempenho de atividades extremamente específicas, encareciam a sua aquisição pelo capitalista. O desenvolvimento de tais máquinas a partir do estabelecimento de padrões de produtos e peças, posteriormente na Segunda Revolução Industrial, caracterizou a tendência de redução desses custos de aquisição e manutenção.

O segundo grupo de máquinas-ferramenta é o das máquinas-ferramenta universais (MFU). Tal como nas anteriores, as MFU eram fruto do trabalho manual de trabalhadores especializados e dele dependiam tanto a manutenção quanto os ajustes de flexibilização que fossem necessários. Existe em grande parte dos processos de produção capitalista, uma necessidade de flexibilização da produção, devido tanto à produção por encomenda, quanto à produção seriada, em que o conjunto dos processos mecânicos de fabricação forma uma base produtiva que deve atender a toda uma gama diversificada de peças necessárias para um dado produto. Conforme Moraes Neto, (2003, p. 29-30)

As máquinas universais, cujo representante mais conspícuo é o torno manual, o qual deve ser considerado uma ferramenta e não uma máquina strictu sensu. Isto porque a função da máquina, desde seu aparecimento e grande difusão nos séculos XVIII e XIX, sempre foi substituir a ação humana no processo produtivo; ora, o torno manual não substitui a ação humana, e sim faz a necessária mediação entre o homem e a matéria, como, por exemplo, o cinzel do escultor.

O autor considera a máquina universal como pertencente ao grupo de máquinas com automação rígida, mas enfatiza que é, a partir destas que, posteriormente, surgiram as

máquinas do terceiro grupo, desenvolvidas para a operação de maneira flexível, ou seja, permitindo a adaptação de diferentes operações para cada variação de produto fabricado. Os sistemas de automação flexível referem-se a um conjunto integrado de sistemas que, tendo em vista a interação avançada entre máquina e homem (operador), em que parte das atividades do operador, mais tarde, são desempenhadas por sistemas de computador, permitem que uma variedade pré-definida de produtos possa ser projetada e manufaturada automaticamente, assumindo trabalho humano, físico e mental, altamente qualificado (CAINARCA et al., 1988). Diferentemente da automação rígida, as máquinas de automação flexível não estão ligadas a um produto específico.

Os efeitos da automação sobre o conteúdo do trabalho, trazem à luz a discussão sobre a Terceira Revolução Industrial, caracterizada pela extrema especialização do trabalho intelectual e pela tendência de apagamento das fronteiras entre este último e o trabalho manual. A automação gradual do trabalho a partir da sua especialização tem como base o avanço e especialização da ciência aplicada para este fim, elevando paulatinamente o grau de subsunção do trabalho no capital. Essa tendência está posta historicamente desde o momento da captura do trabalho qualificado sob a forma artesanal pelo capital, com o surgimento da maquinaria e a Primeira Revolução Industrial. A componente tecnológica, presente em todo o processo de desenvolvimento do capitalismo, adquire mais tarde proporções extremas. A essência do progresso técnico, nesse sentido, é desvendada por Marx nos capítulos d'O Capital sobre Cooperação, Manufatura e Grande Indústria, nos quais Moraes Neto e Bolaño identificam de forma similar uma característica central para o desdobramento do tema. Segundo Moraes Neto (2003, p. 1),

para Marx é fundamental a questão da desqualificação do trabalho ao longo do desenvolvimento tecnológico: no caminho da cooperação simples, com sua prisão à forma artesanal, até a maquinaria, passando pela manufatura, radicalizar-se-ia um movimento de desqualificação do trabalho, o que teria importantes desdobramentos em termos do controle do capital sobre a classe trabalhadora.

Bolaño (2002, p. 7), por sua vez, desenvolve assim a questão:

Em Marx, a idéia de Revolução Industrial remete essencialmente ao processo histórico de passagem da subsunção formal à subsunção real do trabalho no capital. O mesmo ocorre com a chamada Segunda Revolução Industrial, ligada, para o autor, à mesma passagem no último setor da produção material ainda, até aquele momento, organizado sob forma artesanal ou manufatureira: o da produção das próprias máquinas. Nos dois casos, a componente tecnológica do processo é crucial porque permite um avanço da subsunção do trabalho e, conseqüentemente, a expansão da lógica capitalista e da exploração do trabalho vivo. Assim, o elemento principal da Primeira Revolução Industrial foi a máquina-ferramenta e, o da Segunda, a produção de máquinas por meio de máquinas.

Na primeira década do século XIX, o ramo metal mecânico intensificou o uso da maquinaria no setor produtivo, desencadeando efeitos econômicos a partir do aumento da produtividade. Esse ganho de produtividade fora deflagrado pelas inovações que se operavam em torno das máquinas-ferramenta. A Máquina-Ferramenta Universal – MFU, que proporcionou a produção de máquinas por meio de máquinas, é personalizada no torno de descanso deslizante (*slide-rest*), ferramenta que permite a produção de diferentes tipos de peças, considerando que a máquina executa com perfeição os movimentos mais precisos do trabalhador mais hábil ininterruptamente.

Mas, embora as MFU, como máquinas-ferramenta concretizem parte dos movimentos hábeis dos melhores trabalhadores, o processo de trabalho composto por homem e máquina-ferramenta se aproxima bastante das características do trabalho artesanal devido à forte interação e dependência direta do trabalho vivo, característica principal da manufatura. O sistema manufatureiro opera a partir de uma determinada força

de trabalho, composta de trabalhadores individuais que manuseiam uma ferramenta não muito diferente daquela usada no artesanato medieval. O processo de produção pode se apresentar segmentado, tendo como base a divisão de trabalho entre trabalhadores individuais ou em grupos que, divididos em fases de produção, assumem caráter independente conforme as funções que os constituem. O trabalhador é incorporado a um processo de produção, sob o comando do capital, mas para isso, o processo deve se adaptar primeiro ao trabalhador, pois é este que se serve da ferramenta e dita o ritmo do trabalho. Assim, todo o processo de produção está subordinado às habilidades físicas e mentais do trabalhador ou, conforme Moraes Neto (1991, p. 27) “a dependência em relação ao trabalho vivo, enquanto dependência em relação à habilidade do trabalhador manual caracteriza um entrave para o império do capital. É crucial para o modo de produção capitalista se independentizar do trabalho vivo”.

Na manufatura, o ponto de partida revolucionário do modo de produção está centrado na força de trabalho e na grande indústria, o instrumento de trabalho, a partir do aprimoramento das máquinas-ferramenta. Mais importante do que diferença entre a ferramenta e a máquina é a transformação e transição ocorrida desde o ferramental de trabalho manual até a máquina, considerando as suas implicações nas relações sociais e desqualificação do trabalho. A máquina permite ao capital maior poder sobre o trabalho vivo, na medida em que rompe sua dependência direta das habilidades dos trabalhadores. No caso da manufatura heterogênea, no entanto, esse processo é mais complexo.

O trabalho desempenhado pelo homem, em sua interação com as máquinas-ferramenta, como, por exemplo, em sua interação com a máquina-ferramenta universal, não é totalmente desprovido de conteúdo, na medida em que lhe é exigido elevado grau de

habilidade no manuseio e operação da máquina, coexistindo dessa forma, como na manufatura, a hierarquia entre trabalhador especializado e não-especializado. A interação entre homem e máquina-ferramenta, para o qual é vital a destreza do trabalhador hábil, se aproxima assim, em certa medida, do trabalho manufatureiro, mas com enorme aumento da produtividade do trabalho, a partir da precisão de movimentos proporcionados pela máquina aliada à grande flexibilidade do trabalhador.

Assim na maquinofatura a destreza do trabalhador qualificado na manipulação das máquinas-ferramenta se assemelha à dependência em relação ao trabalho vivo no trabalho artesanal e no trabalho qualificado da manufatura, enquanto dependência em relação à habilidade do trabalhador manual. Na manufatura orgânica, que difere da manufatura heterogênea devido à segunda se caracterizar pelo ajustamento mecânico de produtos parciais independentes, a produção está centrada no grupo de trabalhadores qualificados e de ofícios bem definidos com funções parciais que constituem órgãos especializados de um organismo de trabalho. O trabalhador individual ou o grupo de trabalhadores ainda possui algum controle sobre o conteúdo do trabalho, que é exercido conforme a observação do trabalho real, ao invés de ser medido e calculado com base no conhecimento das funções da máquina.

Nesse sentido, as MFU foram um passo importante mas contraditório em direção à automação, devido ao controle do processo produtivo ter forte dependência das habilidades do trabalhador. A relação existente entre o homem e a máquina-ferramenta caracteriza-se como um problema para o capital no processo produtivo, pois por menos especializados que sejam os ofícios, a importância da destreza do trabalhador na sua

interação com a máquina-ferramenta permanece vital. De acordo com Moraes Neto (1995, p. 58) “ao invés de um trabalho aprendicizado, desprovido de qualquer conteúdo, reduzido a pura abstração e supérfluo, temos algo muito mais parecido com o trabalho artesanal [...] Uma MFU não apresenta automação alguma; aliás, representa a própria negação deste conceito.” Nesse sentido a denominação de automação das MFU está ligada mais à precisão dos movimentos orientados pelos trabalhadores do que propriamente a ações automáticas pouco dependentes da intervenção humana.

Independente do conceito de automação, as máquinas-ferramenta desse tipo proporcionaram um inegável incremento na produtividade do trabalho caracterizado pelo uso de ferramentas individualizadas e restritas a ofícios especializados. A produção por meio de máquinas permite a reprodução de forma mais rápida, precisa e fácil do que na manufatura, mais do que isso, a maquinofatura baseada nas MFU representa o início de um novo ciclo no modo de produção capitalista a partir da subordinação do trabalhador ao instrumento de trabalho, mas necessita em grau significativo dos trabalhadores no uso da máquina-ferramenta. Trata-se, portanto, como enfatiza Moraes Neto, de uma forma de organização do trabalho industrial muito diferente daquela própria das indústrias têxtil ou química, que constituem o paradigma da Primeira Revolução Industrial. Assim, a produção mecanizada encontra sua forma mais desenvolvida no sistema orgânico de máquinas-ferramenta combinadas, as quais recebem os seus movimentos de um autômato central, desenvolvendo-se ao longo das diferentes fases do processo de trabalho até a concretização do produto final.

É nesse caso que a maquinaria abre as mais amplas possibilidades ao capital, na medida em que avança mais rapidamente na transformação de trabalho vivo em trabalho

morto e na subordinação daquele à nova forma de organização de trabalho. Essa forma de organização, que se materializa como processo objetivado, gradualmente diminui e restringe a área de atuação do trabalhador, ampliando a dominância do processo de valorização sobre o processo de trabalho, tornando o trabalhador um coadjuvante no processo produtivo, como lembra Palma, (1972, p. 23)

A máquina, como contraposta à ferramenta artesanal, é um mecanismo passível de um processo indefinido que passo a passo conduz à restrição da área de trabalho do trabalhador e, como limite, esvazia de conteúdo esse trabalho. Deste ponto de vista, as operações manuais propriamente ditas não são senão resíduos passíveis de mecanização quando se produzam novas modificações tecnológicas.

Diferentemente da manufatura, a diminuição da área de decisão do trabalho citada por Palma, avança com o estabelecimento de uma nova organização do trabalho a partir da maquinaria, assim como critérios padronizados produtivos e de consumo final menos exigente³. Mas é importante ressaltar que a permanência, no caso da maquinofatura que poderíamos chamar heterogênea, de uma área de decisão do trabalhador delimitada no interior do processo de interação entre homem e máquina-ferramenta caracteriza a inexistência de um processo produtivo completamente automatizado, acentuando, dessa forma, a importância, mesmo que menos contundente que na manufatura, do trabalho vivo. Dessa maneira, a maquinaria, apesar de se apresentar como uma organização de trabalho nova em relação à manufatura conserva ainda em seu interior uma parcela de trabalho de caráter artesanal.

³ “A colonização [Norte] americana começou na costa leste dos Estados Unidos. No Nordeste, isto é, na Nova Inglaterra, concentravam-se as primeiras atividades industriais, enquanto no Sudeste localizaram-se predominantemente as atividades agrícolas. [...] A ‘conquista do Oeste’ foi alicerçada pela construção das estradas de ferro, que rasgaram o país de ponta a ponta e abriram progressivamente novas fronteiras, até alcançar a costa do Pacífico. [...] Do lado da demanda, há a questão da padronização. Observa-se que houve, muito mais do que na Inglaterra, uma elevada propensão no sentido de aceitarem a adotarem elementos padronizados, tanto em termos de consumo produtivo (peças, componentes e equipamentos) como no consumo final.” (TAUILLÉ, 1983, p. 86-87) .

Em todo o caso, a nova organização do trabalho, na área metal-mecânica, caracteriza-se por: (a) mudanças freqüentes advindas dos produtos a serem produzidos e (b) planejamento prévio ou concepção das diversas atividades envolvidas no processo produtivo. Tauile (1983, p. 23-24) caracteriza esses dois aspectos da seguinte forma:

[(a)] devido às freqüentes mudanças do produto de seu trabalho (pequenas séries, lotes e peças sob encomenda), os oficiais mecânicos precisam ter muita destreza manual e experiência prática que se acumulam através do tempo, tornando-se profissionais melhores e mais valorizados. [...] [(b)] Junto à máquina-ferramenta, [os trabalhadores] recebem de seus supervisores diretos os desenhos e instruções e dos serviços de apoio as peças em bruto e as respectivas ferramentas, cames e dispositivos. Interpretam os desenhos, estudam as instruções e revêem o ferramental a fim de verificar se, de acordo com o seu conhecimento prático e sua própria conveniência, devem ser alterados ou corrigidos (...) Após exercer suas habilidades quanto à concepção do próprio trabalho, eles passam efetivamente a executá-lo.

O item (b) descrito por Tauile revela dois aspectos importantes que corroboram para o entendimento do caráter artesanal embutido na maquinaria. “Os serviços de apoio [que fornecem] as peças em bruto” decorrem da existência de uma hierarquia entre trabalhador especializado e não-especializado em que este último assume as tarefas de alimentação do processo produtivo, fornecendo aos trabalhadores especializados os itens necessários para a produção, incluindo, por outro lado, “as respectivas ferramentas”, o que caracteriza assim, o contexto artesanal composto pela virtuosidade do trabalhador especializado e suas ferramentas individuais. O segundo aspecto refere-se a uma importante distinção entre manufatura e maquinaria, pelo fato do trabalho ter sido claramente desmembrado em concepção, no sentido de planejamento prévio das atividades, e execução, que na maquinaria se caracteriza pelas atividades a serem desempenhadas pelo trabalhador diretamente em sua interação com as máquinas. Conforme Braverman (1974, p. 104),

no ser humano, o aspecto essencial que torna a capacidade de trabalho superior à do animal é a combinação da execução com a concepção da coisa a ser feita. Mas à

medida que o trabalho se torna um fenômeno social mais que individual, é possível – diferentemente do caso de animais em que o instinto como força motivadora é inseparável da ação – separar concepção e execução. Essa desumanização do processo de trabalho, na qual os trabalhadores ficam reduzidos quase que ao nível de trabalho em sua forma animal, enquanto isento de propósito e não pensável no caso de trabalho auto-organizado e automotivado de uma comunidade de produtores, torna-se aguda para a administração do trabalho comprado.

Esta separação consciente entre concepção e execução do trabalho foi acentuada na medida em que a divisão capitalista do trabalho foi gradualmente aprofundada, com contingentes de trabalhadores concentrados em um mesmo local de maneira coordenada e articulada, criando um coletivo de trabalhadores integrados de forma orgânica, cuja constituição se assemelha a mecanismos vivos de produção, cujas partes são constituídas por seres humanos.

Todo o trabalho humano, no sentido de atividade que altera o estado natural dos materiais da natureza para melhorar a sua utilidade é consciente e proposital, inclusive o trabalho na manufatura. A separação do trabalho, operada a partir da manufatura e que atinge um maior grau na maquinaria, adquire nesta um formato peculiar. O modo de produção da maquinofatura se estabelece sob uma forma de organização de trabalho composta pelo planejamento prévio das atividades, estabelecimento de instruções e utilização de máquinas-ferramenta. Assim, o modelo manufatureiro difere do maquinofatureiro no sentido em que este último constitui um avanço em direção a um processo de produção mais objetivado. Tal processo se caracteriza fortemente por, diferentemente do modelo manufatureiro, permitir o controle do trabalho vivo pelo capital constante (trabalho morto).

A diferença entre os dois modelos heterogêneos se apresenta justamente no rompimento com os limites representados pela baixa velocidade de multiplicação do

modelo manufatureiro, pois na maquinaria a velocidade do processo é ditada pelo desempenho da máquina. Nesse sentido, o trabalhador passa a responder às necessidades desta ao invés de determinar o ritmo e forma do trabalho. Porém conforme Moraes Neto (1995, p. 58), “encontramo-nos muito distantes do trabalhador aprendicizado à máquina, característica por excelência da máquina segundo a acepção de Marx”. Assim, diferentemente de uma evolução linear, como apresentada por Braverman (1974), segundo Moraes Neto, a partir da maquinaria, a evolução dos processos de trabalho apresenta dois rumos distintos: o taylorismo-fordismo e a automação.

A origem da bifurcação entre taylorismo-fordismo e automação pode ser observada a partir de um elemento comum a ambos os processos que, conforme Braverman, indica que os trabalhadores estão atados aos reais processos de trabalho, sinalizando uma forte interdependência entre trabalho vivo e trabalho morto, a partir da dissociação do processo de trabalho das especialidades dos trabalhadores, separação de concepção e execução e utilização do monopólio do conhecimento para controlar cada fase do processo de trabalho e seu modo de execução. Caracteriza-se o taylorismo, portanto, como controle do trabalho através do controle das decisões que são tomadas no curso do processo de trabalho.

A partir da separação posta na maquinaria entre concepção e execução, Moraes Neto (2003, p. 124) caracteriza o processo produtivo taylorista e posteriormente o fordista, como processos que garantem o avanço da produtividade, através do controle dos tempos e movimentos do trabalhador.

Quem estuda o taylorismo observa que, no fundo, o que Taylor tenta fazer é transformar novamente o homem num instrumento de produção, analogamente ao que se havia tentado na fase pré-maquinaria. Essa visão do homem como

instrumento de produção magnifica-se no fordismo. Ao encetar sua típica inovação, que é a linha de montagem, Ford não fez outra coisa senão coletivizar o taylorismo, com o recurso fundamental da esteira, que procura resolver o problema tipicamente manufatureiro do transporte.

Nesse ponto é importante sinalizar que a organização científica do trabalho conhecida como taylorismo e a criação da linha de montagem fordista, constituem o paradigma de produção taylorista-fordista voltado para a produção em massa. A Figura 1, desenvolvida a partir da interpretação sobre o autor Moraes Neto (1995, p. 83), Palloix (1978, 123-150) e Marx (1980, p. 393, 394 e 440, 441), apresenta a evolução histórica dos processos de trabalho capitalista.

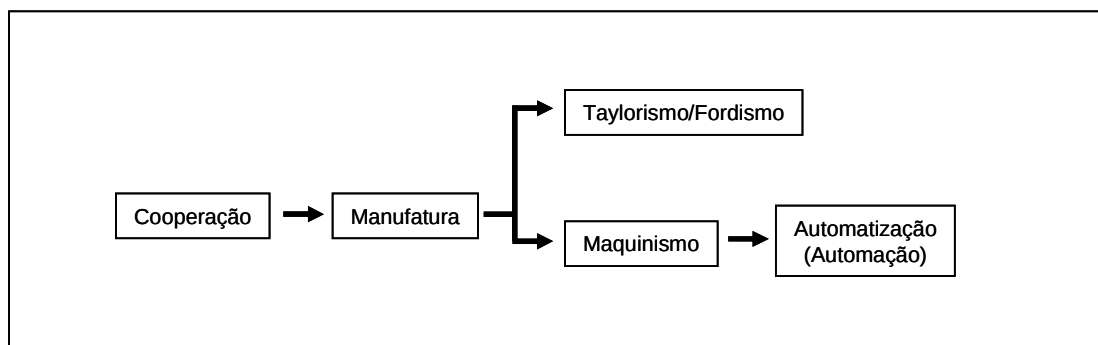


Figura 1 – Representação da evolução do processo de trabalho.

A representação da figura 1 considera que a partir da manufatura ocorre um desdobramento no modo de produção em taylorismo/fordismo e maquinismo. O primeiro tende a transformar o homem em peça de maquinaria, buscando a objetivação do fator subjetivo do processo de trabalho, levando a manufatura a um *desenvolvimento brutal das características próprias do trabalho sob a forma manufatureira* (Moraes Neto, 1995, p. 59). O segundo, decorrente do desenvolvimento das bases técnicas e aperfeiçoamento das ciências, corrige a imprecisão dos movimentos humanos presentes na manufatura, tornando o trabalhador coadjuvante dos processos de trabalho através da introdução da máquina no processo produtivo.

O ramo em direção à automação se coloca a partir da desqualificação de parte do contingente de trabalhadores no sentido de minimizar a atuação desses no processo produtivo e conseguir automatizar a produção em maior escala. Em oposição à desqualificação é requerido pela automação elevada qualificação de parte do contingente de trabalhadores para o papel de preparação e reprogramação das máquinas automatizadas.

A partir da segunda metade do século XX, uma das mais importantes inovações na manufatura, desde a introdução da linha de montagem por Henry Ford, foi o desenvolvimento de máquinas-ferramenta com controle numérico – MFCN⁴. As MFCN são máquinas de automação flexível, característica da produção capitalista contemporânea, que antecede o movimento mais contundente do processo da automação e estabelecem amplas possibilidades de subordinação do trabalho intelectual ao capital.

Diferente da máquina ferramenta universal, que se caracteriza como extensão do corpo do trabalhador no processo de trabalho, exigindo habilidades necessárias ao manuseio adequado do equipamento, David Noble (1989) descreve o surgimento das MFCN a partir do estudo de engenharia que permitiu a introdução de computadores no controle direto das máquinas-ferramenta para a fabricação de peças para aeronaves para a força aérea dos Estados Unidos. A partir do estudo, foi possível o desenvolvimento de uma máquina fresadora vertical que, a partir de movimentos pré-programados com o

⁴ Durante esse intervalo de tempo, o fato mais inovador talvez tenha sido a utilização de linhas *transfers* na produção de motores de automóveis, sem que isso constituísse, no entanto, uma inovação radical em termos de técnicas de manufatura. Uma linha *transfer* é composta por uma sequência de máquinas articuladas por mecanismos de transferência de peças de uma para outra máquina, automaticamente. Nela são usinadas praticamente todas as peças do motor de um automóvel. Dada a rigidez da base técnica eletromecânica, só fazia sentido em utilizar linhas *transfer* quando a quantidade a ser produzida de um determinado modelo de motor fosse suficiente para produzir economias de escala (TAUILE, 1983).

auxílio de computador⁵, permitia movimentos extremamente precisos na produção das peças, além de exigir por parte dos trabalhadores novas habilidades manuais de intensa qualificação para o manuseio do novo equipamento. A política industrial e tecnológica do governo norte americano contribuiu inicialmente para a produção das MFCN em maior escala, a partir do financiamento da compra, instalação e manutenção de mais de 100 máquinas fresadoras com controle numérico nas empresas sub contratadas da força aérea estadunidense (TAUILE, 1983).

Nas MFCN, o conhecimento dos trabalhadores é explicitamente tangibilizado na forma de programas de computador que descrevem todos os movimentos a serem seguidos pela máquina, ocasionando um aprofundamento da subsunção do trabalho, com a transferência da concepção do trabalho para os escritórios, na forma de linguagem matematizada. De acordo com Noble, não se pode saber a priori qual será a conformação e o uso final que esta máquina ou organização do processo de trabalho receberá quando “chegar” no chão de fábrica. A relação de causa e efeito nunca é automática – ou a causa é a tecnologia ou as escolhas sociais por trás dela – mas é sempre mediada por um complexo processo cujo resultado depende, em última análise, da força relativa das partes envolvidas. Como consequência, os efeitos não estão em consonância com as expectativas implícitas nos desenhos originais (Noble, 1989).

O trabalho constituído pelas MFCN é de elevada complexidade e exige do trabalhador um alto grau de habilidade tanto na execução quanto na supervisão da tarefa. Tais habilidades se caracterizam como conhecimento acumulado que, gradativamente,

⁵ Programação efetuada de maneira bastante restrita, devido às limitações computacionais do período. Para se ter uma idéia de tal limitação, basicamente não existia *software* e as máquinas eram integralmente programadas a partir do próprio *hardware*.

permite ao capital produtivo, baseado na ciência aplicada, evoluir para processos produtivos cada vez mais objetivados.

Em 1947, John Parsons, da Parsons Corporation, iniciou experimentos com a idéia de utilizar dados de curvatura de 3 eixos (coordenadas x, y e z que representam as três dimensões no espaço: altura, largura e profundidade) para controlar movimentos de ferramentas em máquinas para a produção de componentes para a indústria de aviões. Em 1952, o comando numérico ficou pronto e demonstrou que movimentos simultâneos com 3 eixos eram possíveis, usando um controlador construído em laboratório e um eixo vertical. Em torno de 1955, depois de alguns refinamentos, o comando numérico tornou-se disponível para a indústria (Machado, 1989). Os primeiros comandos numéricos eram programados a partir de fitas ou cartões perfurados, devido às limitações tecnológicas existentes no período. Devido ao tempo e esforço necessários para editar programas em fitas, posteriormente computadores foram introduzidos para auxiliar a programação. Historicamente, conforme apresentado na Figura 1, o conceito de automação estaria mais bem empregado com o processo desencadeado a partir do maquinismo, mais especificamente com o surgimento das MFCN. Com o desenvolvimento da ciência aplicada, altamente especializada, à luz das inovações produtivas, em direção à microeletrônica, o desenvolvimento da MFCN culminará no surgimento do Controle Numérico Computadorizado – CNC. Segundo Moraes Neto (2003, p. 44-45),

A incorporação da microeletrônica ao nível das máquinas-ferramenta da metal-mecânica reflete um duplo movimento:

- a) Ruptura radical de um status teórico de profundo atraso no relacionamento homem-máquina, (como é o caso da MFU), em direção a um status teórico de máquina em sua forma mais avançada (como é o caso da Máquina-Ferramenta com Controle Numérico – MFCN). Em outras palavras, uma máquina automatizada no mais alto grau e bastante complexa substitui, de forma abrupta, uma ‘máquina’ (se é que podemos chamá-la assim,

rigorosamente falando) extremamente atrasada, altamente dependente da habilidade do trabalho vivo.

b) Igualdade teórica da nova máquina [MFCN] com a máquina de automação dedicada, no sentido de que ambas se enquadram perfeitamente no conceito de automação.

Com o aperfeiçoamento da microeletrônica foi possível a transição do Controle Numérico das MFCN para o Controle Numérico Computadorizado. O Controle Numérico Computadorizado é um equipamento eletrônico incorporado à máquina-ferramenta que permite maior flexibilidade, precisão, uniformidade e conseqüentemente formas mais desenvolvidas e complexas da indústria mecânica. Para Kaplinsky (1885), a diferença crucial entre o comando numérico e o comando numérico computadorizado é que este último oferece maior flexibilidade ao trabalhador programador, devido ao fato de permitir a interrupção do trabalho da máquina e sua reprogramação diretamente no chão de fábrica (*shop-floor*). Tais fatores acentuam o processo de desqualificação do trabalho na medida em que a tarefa de (re) programação é feita inclusive de forma remota, fora do chão de fábrica, e desvinculada em relação à tarefa de supervisão. Com a introdução do *microchip* a partir da década de 1970, ocorre a difusão de uma nova geração de equipamentos, oriundos de uma evolução das MFCN. A capacidade de processamento de informações do *microchip* tornou possível a programação de comandos a serem operados, na própria máquina. As MFCNC ou CN simplesmente⁶, antigas MFCN, passaram a ser utilizadas em outros ramos da indústria, tal como o automobilístico e o de bens de capital. As MFCNC pertencem a um grupo de máquinas que, dotadas de dispositivos de controle microeletrônicos, são capazes de desempenhar atividades de produção com maior precisão do que o mais hábil dos trabalhadores.

⁶ As máquinas de Controle Numérico Computadorizado são chamadas de CNC por Moraes Neto (2003) e de Máquinas-Ferramenta de Controle Numérico Computadorizado - MFCNC por Tauile (1983).

Outros tipos de equipamentos similares às MFCNC incluem os *Direct Numerical Control* – DNC e os Sistemas Flexíveis de Manufatura – SFM ou *Flexible Manufacturing System* - FMS. O FMS é um sistema de controle centrado no computador, formado por duas ou mais máquinas de controle numérico computadorizado equipadas com um sistema robotizado para transferência automática de peças de uma máquina para outra, também com capacidade de operar por longas horas sem qualquer interferência humana. O computador central controla a produção desde o suprimento de material até a finalização do processo, de acordo com um programa armazenado na memória da máquina. Em outras palavras, o FMS executa a usinagem de uma larga variedade de peças, em pequenos lotes, durante horas ininterruptas, sem a interferência humana na operação, e em seqüência de produção programada (SIMHON, 1986). Nas máquinas de controle numérico, uma vez definida a sua programação pelos programadores, estas se apresentam prontas para serem operadas no chão de fábrica. As máquinas com comando numérico computadorizado permitem toda flexibilidade da tecnologia anterior além de poderem ser reprogramadas diversas vezes, tanto pelo programador, quanto pelo programador/operador, no chão de fábrica.

Até ser introduzido o CNC, a automação incorporada às máquinas se caracterizava por reproduzir ou aprisionar a habilidade do trabalhador em movimentos mecânicos precisos, além de flexibilizar a produção. Uma vez programadas, as máquinas executariam precisamente os movimentos necessários para que o resultado da sua produção fosse contínuo e padronizado. Esta última característica faz com que o trabalhador perca para a máquina, o controle do processo produtivo e conseqüentemente a autonomia que detinha sobre o processo de produção. O controle do processo produtivo pelo capital soluciona em parte a necessidade do controle dos tempos e movimentos que vem a ser o cerne da

problemática do processo produtivo taylor-fordista. A bifurcação do maquinismo em taylorismo-fordismo e automação se dá pelas formas de controle do trabalho vivo. No taylorismo, o capital ao invés de substituir o trabalho vivo por trabalho morto – como na automação – procura dominar o trabalho vivo em si mesmo, “obrigando o trabalhador a trabalhar com uma regularidade de uma peça de maquinaria” (MARX, 1980, p. 293). Ou seja, o taylorismo, conforme Moraes Neto (2003, p. 34), busca “objetivar o fator subjetivo, o trabalho vivo”. Esta última colocação ganha extrema importância na análise da fase de concepção envolvida no processo produtivo automatizado, mais especificamente com a introdução das máquinas de Controle Numérico Computadorizado no processo de trabalho.

O controle do trabalho pelo capital através do controle das decisões que são tomadas no curso do trabalho está associado ao paradigma de produção em massa, com maquinaria desenvolvida para a reprodução em série e controle objetivado do trabalho vivo. Até este ponto, a evolução do processo produtivo, desde a manufatura até a bifurcação em taylorismo-fordismo e automação, fora descrito sob o ponto de vista meramente tecnológico, ou seja, a partir do acúmulo de conhecimento desenvolvido com o apoio das ciências, que reestrutura o modo de produção capitalista de forma contundente, aumentando as assimetrias traduzidas em profundas diferenças de especialidades entre trabalhadores dentro de um mesmo processo de trabalho. Ambos os processos produtivos, taylorismo-fordismo e automação, estão respectivamente associados à produção em massa e à produção flexível, tendo o primeiro criado uma reserva de trabalhadores que se organiza em classe para reclamar os ganhos proporcionados por este modo de produção. Bolaño (2002, p. 6) sugere que

assim, as tecnologias informacionais aplicadas à produção e a robotização, além de se traduzirem de imediato numa redução importante dos postos de trabalho, são adotadas de acordo com o novo paradigma da produção flexível, dirigindo a produção a segmentos de consumidores, à diferença do paradigma da massificação do período do fordismo. Assim, coletivos reduzidos de trabalhadores, com alta qualificação, produzem para segmentos específicos da população, produtos diferenciados. [...] Nunca é demais lembrar que segmentação e massificação, homogeneização e diferenciação, antes de opor-se radicalmente, complementam-se.

A segmentação citada pelo autor aparece como pano de fundo no processo de automação e caracteriza-se como catalisador do processo de subordinação do trabalho intelectual que se opera a partir daí.

A introdução do Controle Numérico Computadorizado abre um novo leque de possibilidades ao capital no momento em que possibilita a subordinação dos tempos e movimentos e tomadas de decisões no interior do processo de trabalho, simplificando as atividades do trabalhador a partir da degradação das suas qualificações profissionais. O processo de deslocamento e transferência de tomada de decisão para o nível gerencial se concretiza a partir das programações realizadas nas máquinas por parte dos programadores. A separação entre a execução e a concepção do produto está explicitamente posto por Taylor (1912, p. 98-99) em seu estudo sobre a gerência científica do trabalho. “Todo possível trabalho cerebral deve ser banido da oficina e centrado no departamento de planejamento ou projeto (...)”. De forma mais contundente, com a manufatura ocorre a desumanização do processo de trabalho a partir da crescente separação entre a concepção e execução, causando aos trabalhadores a isenção de propósito do trabalho. Braverman (1974 p. 104) afirma que

esse poderia ser chamado o princípio da separação entre concepção e execução, melhor do que seu nome mais comum de princípio da separação do trabalho mental e manual (embora semelhante ao último e, na prática, quase sempre idêntico). Isso porque o trabalho mental, trabalho principalmente do cérebro, é também suscetível de separação de concepção e execução conforme o mesmo princípio: o trabalho

mental é primeiro separado do manual e (...) depois subdividido rigorosamente de acordo com a mesma norma.

Esta última característica citada pelo autor, a partir da rigorosa subdivisão do trabalho intelectual, assume extrema complexidade uma vez que o controle do processo se transfere para a fase de concepção, ou seja, para a etapa de programação da máquina.

Se, por um lado, as MFCN simplificam ou restringem as atribuições do trabalhador, por outro, exigem maior grau de especialidade, na medida em que se disseminam os equipamentos CNC, pois o trabalhador incorpora as funções de operador e programador da máquina e define o raciocínio lógico que permite a esta adaptar-se automaticamente a mudanças durante a execução do trabalho, seguindo roteiros pré-definidos cada vez mais complexos. Assim, se no processo da maquinofatura as decisões de execução da produção, tais como a forma e detalhes específicos de utilização da máquina para a produção, deriva das virtuosos do trabalhador, a partir das máquinas CNC, essas são controladas pela máquina. Tais decisões são pré-programadas na etapa de concepção do trabalho para então serem executadas de maneira assistida e monitorada. A automatização da produção diminui a necessidade de supervisão da tarefa no chão de fábrica e aumenta a complexidade e necessidade da concepção e planejamento da mesma. Conforme observado no estudo realizado por Elenice Leite (1986), percebe-se um movimento de integração entre perfis, em que os postos de trabalho, inicialmente separados, de programador, supervisor e operador convergem para um perfil único programador/supervisor/operador. Neste sentido, há uma desqualificação do trabalho na medida em que não se faz necessária a existência de um perfil exclusivo de “operador *stricto sensu*”, ou seja, de operador que só opera (MORAES NETO, 2003, p. 48), aliado

ao fato de que o novo perfil integrado (programador/supervisor/operador) necessita de virtuosos à altura da complexidade de operação de máquinas como as MFCN.

As máquinas com controles numéricos computadorizados proporcionaram ao capital, a ampliação da sua capacidade de capturar e subordinar o trabalho vivo, ampliando os limites da automação. Quais os limites dessa captura e de que forma ela se concretiza? Para responder a esta pergunta, é preciso ter em mente, conforme Bolaño (2002, p. 7) que “a componente tecnológica do processo é crucial porque permite um avanço da subsunção do trabalho e, conseqüentemente, a expansão da lógica capitalista e da exploração do trabalho vivo”. Em Moraes Neto (1995, p. 58), a automação aparece historicamente, a partir da microeletrônica, com aperfeiçoamento do modelo maquinofatureiro composto pela unidade homem-máquina.

[...] somente através do desenvolvimento da microeletrônica foi possível quebrar o esquema ‘automação dedicada/não-automação flexível’. Isso porque, sabidamente, a microeletrônica permitiu a geração de equipamentos que passaram a ter, em alguma medida, faculdades antes monopolizadas pelo homem.

A automação dedicada é caracterizada por ferramentas desenvolvidas para a elaboração de um único tipo de peça ao longo da sua vida útil. A não-automação flexível de que fala o autor nada mais é do que o modo de produção composto pela unidade homem-máquina que opera com relativa limitação se comparado com o processo produtivo composto pela automação *stricto sensu*. A não-automação flexível possui uma forte dependência da habilidade do trabalhador que, nesse sentido, figura ainda como principal ator dos processos de trabalho de produção em massa. Segundo Moraes Neto, tal dependência fora observada no taylorismo e “corrigida” no fordismo com a objetivação do trabalho vivo a partir do estudo da gerência científica. Com o desenvolvimento da microeletrônica e o surgimento da automação flexível, o trabalhador perde, de fato, o

lugar de principal ator do processo de trabalho, tornando-se coadjuvante e conseqüentemente supérfluo. Esse deslocamento é dado pela componente tecnológica, que permite a transferência, mais uma vez, do conhecimento do trabalho vivo para o trabalho morto, no mesmo sentido da “acumulação primitiva de conhecimento” (BOLAÑO, 2000, p. 9), mas desta vez em relação às funções intelectuais que o operário típico do taylor-fordismo ainda dominava. Essa “subsunção do trabalho intelectual” só é possível, como aponta Bolaño (1995), pelo desenvolvimento das tecnologias informacionais. O avanço dessas tecnologias proporcionou o desenvolvimento das linguagens computacionais que permitem a incorporação às máquinas de “faculdades antes monopolizadas pelo homem” (MORAES NETO, 1995).

Dessa forma, intensifica-se o caráter excludente do processo produtivo, pois a flexibilidade exige máquinas e postos de trabalho diferenciados, em uma mesma linha de produção, ocupadas, uns por trabalhadores altamente especializados, capazes de reprogramá-las a partir de linguagens computacionais, e os outros, por trabalhadores desqualificados necessários apenas para a alimentação das máquinas com matéria-prima e o transporte das mercadorias produzidas.

As amplas possibilidades abertas ao capital se estabelecem na medida em que, diferentemente dos processos anteriores, o controle numérico computadorizado pressupõe não apenas o controle da execução, mas da própria concepção. A concepção das máquinas com componentes numéricos computadorizados se traduz na avaliação prévia, por parte dos programadores, do espectro de decisões a serem tomadas pela máquina a fim de serem programadas com a finalidade de executar com flexibilidade a peça ou produto a ser produzido. Uma única máquina pode ser programada de diversas maneiras com a

finalidade de atender às demandas da produção flexível. A subordinação renovada do trabalho vivo proporcionada pelo avanço da microeletrônica, neste caso de uma parcela intelectual do trabalho, segue o mesmo rumo da acumulação de conhecimento ocorrida na fase prévia à primeira Revolução Industrial: as bases de concepção e execução, que compõem a organização do trabalho, apresentam-se a partir do desenvolvimento do controle numérico computadorizado de maneira diferenciada em relação à forma assumida no taylorismo-fordismo. Diferentemente deste, a automação não objetiva transformar o homem num instrumento de produção. Ao contrário, extrai deste o conhecimento necessário para desenvolver e inserir no processo de trabalho um instrumento capaz de executar integralmente as suas ações a partir de um processo de apropriações do conhecimento. Tal processo, mais do que contínuo, apresenta-se como gradual, no sentido de que a máquina assume paulatinamente as funções exercidas pelo trabalhador, excluindo-o gradativamente do processo de trabalho, com acentuação das assimetrias.

O programador das máquinas CNC, através de uma linguagem de computador composta de comandos pré-estabelecidos, descreve e programa na máquina os comandos decisórios que deverão compor as tarefas desempenhadas por esta, de forma que o operador precise apenas supervisionar o processo, alimentando a máquina com matéria-prima. No processo produtivo composto por máquinas de controle numérico computadorizado, a fase de concepção assume caráter parcialmente codificado no sentido de que o trabalhador não dispõe de total liberdade para conceber o trabalho. Precisa primeiro conhecer uma linguagem de programação capaz de traduzir a sua concepção em comandos para a máquina. Segundo Papert (1998, p. 34), “programar significa, nada mais, nada menos do que se comunicar com o computador”. Mas o ato de programar a máquina está inserido em um contexto de desenvolvimento e produção definido e pré-determinado,

que extrapola o conceito de comunicação e interface entre homem e máquina, caracterizando-se como trabalho conceitual codificado a serviço do capital produtivo⁷.

Nessa perspectiva, Bolaño (2002) fala da passagem da subsunção formal à subsunção real do trabalho intelectual, seguindo uma linha de raciocínio similar à de Marx na sua análise da passagem da manufatura à grande indústria. Segundo Bolaño, a Terceira Revolução Industrial pode ser vista também como um processo de avanço da subsunção do trabalho (intelectual) ligado a uma intelectualização geral de todos os processos de trabalho e do próprio consumo. No seu trabalho inicial sobre o tema, Bolaño (1995) destaca o papel do *software* como conceito geral, sem, no entanto, aprofundar a questão específica das linguagens e métodos computacionais que permitem o amplo desenvolvimento e produção dos *softwares*.

Nesse processo, a subsunção do trabalho e a crescente incorporação do conhecimento científico realimentam a acumulação capitalista que, contraditoriamente, acentua a exclusão do homem do processo produtivo, automatizando inicialmente os seus movimentos e, em seguida, com o desenvolvimento das ciências aplicadas e o aperfeiçoamento tecnológico, especialmente a partir da Segunda Revolução Industrial, as próprias decisões a serem tomadas ao longo dos processos do trabalho. Moraes Neto apresenta o taylorismo como processos de trabalho que, embora sobrevivam no capitalismo monopolista, não correspondem às tendências dominantes do avanço tecnológico (MORAES NETO, 1991), pois enquanto o sistema de máquinas é a base

⁷ Nesse sentido, Sohn-Rethel (1989, p. 14) ressalta as características do processo de trabalho com base em quem concebe ou se apropria da concepção intelectual, pois deve-se distinguir, se o fim almejado de um processo de trabalho se encontra idealmente na cabeça daquele que leva adiante o trabalho, ou nas cabeças de vários, que realizam conjuntamente o trabalho, ou então em uma cabeça estranha, que envia aos trabalhadores só partes divididas do processo, as quais de forma alguma significam uma finalidade pretendida, porque para os executores elas são postas por outros.

técnica adequada à valorização do capital, o taylorismo molda o trabalho humano à imagem do maquinismo. Na forma mais desenvolvida do taylorismo, o fordismo, se aprofunda a transformação do homem em máquina, limitando os tempos e movimentos deste com o intuito de transformar o contingente de trabalhadores num sistema de homens objetivados segundo as necessidades da valorização do capital. Nesse sentido, Moraes Neto (1991) apresenta essa forma de trabalho como uma forma de manufatura “maquiada”, interpretada erroneamente por alguns autores como uma forma moderna de manufatura. Para o autor, essa “neomanufatura” consiste na transformação do trabalhador em máquina, de modo que a gerência científica tenta a todo tempo controlar o do trabalho vivo através do domínio dos tempos e movimentos, não havendo, portanto direcionamento à automatização. Inicialmente a mecanização da indústria metal-mecânica ocorre no transporte de peças entre os trabalhadores, mais precisamente nas esteiras rolantes que interligam os grupos de trabalhadores, reduzindo assim o tempo de transporte na linha de montagem. Para o autor “a máquina molda, costura e solda, mas não monta, a não ser que surja um novo tipo de máquina, filha do desenvolvimento da microeletrônica – o robô. E com esse tipo de máquina, a linha de montagem, reinvenção da manufatura, ajusta-se, abruptamente, ao princípio da maquinaria” (MORAES NETO, 1991, p.12-13).

Bolaño (1995), por sua vez, aponta o desenvolvimento das tecnologias informacionais como resultado da “subsunção do trabalho intelectual”, possível somente devido à transferência do conhecimento do trabalho vivo para o trabalho morto, como “acumulação primitiva de conhecimento”, em relação às funções intelectuais que o operário típico do taylor-fordismo ainda dominava. Essa subsunção do trabalho intelectual só é possível pelo desenvolvimento das tecnologias informacionais, que permitem ao capital um avanço no domínio do trabalho intelectual do trabalhador, trazendo

consequências tanto nas decisões automatizadas tomadas pelas máquinas no chão de fábrica, quanto no desenvolvimento e uso das linguagens computacionais que permitem a programação e controle dos *softwares* que operam tais máquinas.

Assim, o capital adquire condições que lhe garantem tanto a reinvenção da manufatura, no sentido daquele “ajuste abrupto ao princípio da maquinaria”, de que falava Moraes Neto quanto a automatização de processos intelectualizados, como no caso das linguagens computacionais, que permitem a criação de *softwares* por meio de *softwares*.

Aqueles processos específicos que garantem a progressiva concretização da codificação do conhecimento são a forma contemporânea dos processos de subsunção. No período manufatureiro, a forma artesanal fora desmembrada em atividades menores a fim de ser melhor compreendida e explorada pelo capital, o que culminou com o desenvolvimento de ferramentas padronizadas, que deram origem às máquinas capazes de garantir a subsunção real, ao desempenhar as atividades anteriormente desenvolvidas individualmente. O ato de programar a máquina através do Controle Numérico Computadorizado é um ato de codificação do conhecimento, que exige o estabelecimento de linguagens e métodos computacionais específicos, os quais permitem a subordinação de tipos determinados de trabalho intelectual. As linguagens e métodos computacionais para desenvolvimento de *software*, concretizam o processo de subsunção do trabalho intelectual a partir da perda do controle e autonomia, por parte do trabalhador sobre o processo de produção. Nesse contexto o processo de desenvolvimento da produção de *software*, como bem intuiu Bolaño (1995), ou das TIC, como insistirá posteriormente, é a chave do problema. O interessante é que essa produção será também parcialmente

subsumida, no sentido de que o conhecimento dominado pelo capital será parcialmente codificado.

A programação das máquinas pressupõe a existência de máquinas automatizadas que incorporam os movimentos de trabalhadores hábeis e, considerando as máquinas de Controle Numérico Computadorizado, decisões de execução pré-ordenadas sob a forma de comandos computadorizados. Tais comandos são estabelecidos a partir da descrição das decisões de execução das tarefas com base em linguagens computacionais que permitem a comunicação entre o homem (programador) e a máquina possuidora da unidade de Comando Numérico Computadorizado. O processo de estabelecimento das linguagens e métodos computacionais que permitem o amplo desenvolvimento e produção de *softwares* é o elemento central da intelectualização geral dos processos de trabalho na Terceira Revolução Industrial.

Nesse sentido, é necessária a verificação dos limites da subsunção do trabalho com base na análise dos limites à codificação do conhecimento no processo produtivo a partir da verificação das linguagens e métodos computacionais. O processo produtivo e conseqüentemente os produtos por ele gerados podem ser separados nas fases de concepção e execução, sendo que esta última encontra-se em processo de codificação e automação extremamente acentuado, enquanto a primeira, em pleno processo de subsunção. O processo que se estabelece historicamente a partir do início da Terceira Revolução Industrial é similar àquele ocorrido na Primeira, mas em bases científicas e de automação extremamente desenvolvidas, conseqüência dos avanços decorrentes dos desdobramentos da Segunda ao longo de todo o século XX. O conceito de linguagem

computacional e seu desenvolvimento histórico concreto é fundamental. A linguagem computacional como ferramenta nos processos de trabalho contemporâneos deve ser considerada sob dois pontos de vista. Primeiro, como ferramenta no sentido de instrumento inacabado e em processo de aperfeiçoamento que é fornecido ao trabalhador para que este desempenhe as suas atividades, como por exemplo, a produção industrial por intermédio das ferramentas CAD/CAM. Segundo, como instrumento de trabalho propriamente dito, inserido como parte integrante do processo produtivo de *software* onde assume o papel de pré-determinar as ações do trabalhador, o que garante a subsunção do trabalho intelectual.

O ramo de linguagens utilizadas para a programação das máquinas CNC tem a finalidade de facilitar a programação das máquinas por parte dos trabalhadores e incorporar descrições pré-estabelecidas para a especificação dos comandos a serem desempenhados na execução das tarefas. O processo produtivo composto por programador, máquina CNC e operador pode ser melhor definido nos elementos programador, linguagem computacional, unidade de controle, máquina e operador, pois a linguagem computacional se apresenta como uma importante ferramenta na organização do trabalho composta pelos elementos descritos.

O processo produtivo pode, assim, ser dividido em duas partes. Na primeira, composta pela máquina CNC e o operador, repousa todo o processo de subsunção do trabalho ocorrido desde a sua forma artesanal até a automação. O trabalhador, neste caso o operador, assume o papel de coadjuvante no processo de trabalho por suas habilidades manuais terem sido transferidas quase que integralmente para a máquina. Na outra parte,

composta por programador e linguagem computacional, o processo de subsunção do trabalho intelectual se apresenta inacabado, por conta da subjetividade envolvida na tarefa de programar a máquina para a execução de decisões pré-determinadas.

CAPÍTULO 2

LINGUAGENS E FERRAMENTAS COMPUTACIONAIS

As MFCN, são máquinas extremamente complexas no concernente às possibilidades abertas ao capital pois não mais congelam as habilidades mecânicas do trabalhador mais hábil em sua estrutura, mas num programa de computador que pode ser reprogramado, dependendo das necessidades específicas da tarefa. Se nas MFU a habilidade (*skill*) do trabalhador era exigida com mais evidência, a partir das MFCN esse trabalho se baseia intensivamente no conhecimento (*knowledge*), o qual é definido por Kaplinski (1985, p. 435) como

[...] a abrangência do entendimento de um processo ou informação a um nível abstrato, tais como aqueles que podem ser transmitidos a outro indivíduo de forma igualmente abstrata. Como tal, o conhecimento deve ser explicitamente racionalizado em termos abstratos que possam ser prontamente entendidos – um processo que passamos a conhecer como ciência e tecnologia. Skill compreende um conjunto de experiências exercitadas, que pode envolver não apenas a aquisição do conhecimento, mas também um grau maior ou menor de aptidão natural e regras implícitas de operação. Skills são adquiridos individualmente e envolvem a combinação de aprendizagem abstrata, aptidão e experiência, mas o mesmo não é verdadeiro para o knowledge, que é essencialmente abstrato e menos individualizado.

O processo de desenvolvimento de *software* estabelece métodos aliados a um conjunto de atividades, parcialmente ordenadas, com a finalidade de obter o produto final, ou seja, um produto de *software*. Tal processo é estudado na área de Engenharia de *Software*, que visa assegurar o cumprimento das fases de desenvolvimento do *software* com o intuito de atingir os objetivos do projeto. Na década de 70, como consequência do desenvolvimento das ferramentas de projeto assistido por computador (*Computer Aided Design* – CAD), surgiram as primeiras ferramentas conhecidas como ferramentas de engenharia de *software* assistida por computador (*Computer Aided Software Engineering*

– CASE), que abrangem toda ferramenta baseada em computadores que auxiliam atividades de engenharia de *software*, desde a concepção do projeto do *software*, passando pela análise de requisitos e modelagem, até a programação e testes finais. As ferramentas CASE, na prática, têm como objetivo automatizar atividades de pré-codificação ou concepção. Para isso, definem uma série de diagramas e regras para moldar a forma como o indivíduo deve pensar e conceber o *software*. A engenharia de *software* é concebida de forma a capturar a concepção humana com o intuito de codificação do conhecimento. PFLEEGER (2004, p. 10) sugere que

assim como em outras áreas, em uma abordagem de engenharia de software, inicialmente o problema a ser tratado deve ser analisado e decomposto em partes menores, em uma abordagem “dividir para conquistar”. Para cada uma dessas partes, uma solução deve ser elaborada. Solucionados os problemas isoladamente, é necessário integrar as soluções. Para tal, uma arquitetura deve ser estabelecida. Para apoiar a resolução de problemas, procedimentos (métodos, técnicas, roteiros etc) devem ser utilizados, bem como ferramentas para parcialmente automatizar o trabalho. Neste cenário, muitas vezes não é possível conduzir o desenvolvimento de software de maneira individual. Pessoas têm de trabalhar em equipes, o esforço tem de ser planejado, coordenado e acompanhado, bem como a qualidade do que se está produzindo tem de ser sistematicamente avaliada.

Os *softwares* que compõem o comando numérico computadorizado das máquinas automatizadas são programados a partir das linguagens computacionais por programadores que transformam em código o conhecimento extraído dos trabalhadores de um determinado setor. Trata-se, portanto, de uma forma de trabalho morto. Mas o *software* tem uma especificidade em relação à máquina, pois é passível de ser não apenas reprogramado pelos operadores, na máquina CNC, mas inclusive completamente reestruturado pelos programadores, em ambos os casos, tantas vezes quanto necessário aos ajustes dos processos de trabalho. Essa particularidade do *software* se deve a sua função de mediação entre a máquina e o homem.

O estudo realizado por Elenice Leite (1986) em 19 empresas paulistas a respeito das mudanças ocasionadas pela introdução das MFCN no processo produtivo industrial, ilustra a existência de uma separação no trabalho dos operadores e programadores dessas máquinas. O estudo revelou que o processo de desqualificação do trabalho implica também em sua intensa especialização: funcionários altamente qualificados responsáveis pelo planejamento do trabalho, em contraste com os operadores de máquina responsáveis pela sua execução. Baseado em uma abordagem qualitativa exploratória e apesar de não comportar generalização nos resultados, o estudo permite, em todo caso, observar algumas conseqüências da introdução das MFCN no processo produtivo conforme a adoção das máquinas-ferramenta, vantagens e desvantagens ocorridas, formação profissional e principalmente do perfil dos novos “profissionais do Controle Numérico”:

Em relação aos operadores de máquinas, há dois tipos de alterações: de um lado, como se sabe, a MFCN praticamente ‘assume’ as tarefas de execução da peça, simplificando ou restringindo as atribuições do operador; de outro, amplia e torna mais complexas as tarefas de preparação da máquina, passando a exigir novos conhecimentos e habilidades do mesmo. Essa dupla tendência com relação ao perfil dos operadores CN – simplificação de um lado, e maior complexidade do outro – decorre da manifesta preferência, por parte das firmas, pelo profissional mais completo, capaz de manejar os novos (e caros) equipamentos com maior segurança, independência e responsabilidade. Há somente duas firmas em que predomina o perfil do ‘operador que só opera’, complementado pelos ‘preparadores de MFCN’. Nas demais, o ‘operador/preparador’ já é ou tem perspectivas de tornar-se a categoria dominante, podendo, no futuro, transformar-se em ‘operador/preparador/programador’ à medida que se disseminam equipamentos de geração CNC.

Neste sentido, Cainarca et al. (1988, p. 82) definem os sistemas de automação flexíveis como

[...] um conjunto integrado de sistemas que, agregando *hardware* e *software* de forma avançada, permitem uma variedade de produtos pré-definidos serem projetados e manufaturados automaticamente, assumindo atributos humanos físicos e mentais de maneira antes não concebida. No presente, tais sistemas estão no primeiro estágio de introdução de um processo de difusão, onde soluções redundantes são constantemente geradas, competindo intensivamente com outras semelhantes e inseridas na estrutura dos setores de suprimento e demanda.

Neste contexto, visualiza-se a real possibilidade aberta ao capital. No processo produtivo automatizado, as máquinas CN e CNC não mais consolidam as habilidades extraídas do trabalhador, conforme intensificado na relação do trabalho com as MFU. Mais do que isso, aliadas aos produtos de *software* desenvolvidos especificamente para trabalhar em conjunto com as mesmas, ampliam e intensificam significativamente a subsunção do trabalho vivo no processo produtivo, conforme a sua flexibilização e a ampliação do controle sobre a concepção e execução do trabalho.

A Figura 2 apresenta duas formas distintas de aplicação de produtos de *software*. A primeira forma é o CAD/CAM, que auxilia a concepção (*design*) e criação (execução) de peças piloto (que uma vez definidas serão produzidas continuamente de forma automatizada nas máquinas CNC) através de uma interface computadorizada. O *software* exerce o papel de mediação entre homem e máquina em que aquele armazena conhecimento acumulado capaz de permitir ao sujeito a fabricação de peças com maior grau de precisão em comparação com a maquinofatura. O termo CAD (Computer Aided Design - Projeto Assistido por Computador) pode ser definido como sendo o processo de projeto que se utiliza de técnicas gráficas computadorizadas, através da utilização de programas (*software*) de apoio, auxiliando na resolução dos problemas associados ao projeto. Os sistemas CAD - (Computer Aided Design) e CAM – (Computer Aided Manufacturing), respectivamente Projeto Auxiliado por Computador e Manufatura Auxiliada por Computador, são ferramentas que desempenham um papel fundamental

para a viabilização de um projeto em tempos reduzidos, oferecendo oportunidade para simular e reduzir custos na fase de desenvolvimento do produto⁸.

A tecnologia CAD/CAM corresponde à integração das técnicas CAD e CAM num sistema único e completo. Isto significa, por exemplo, que se pode projetar um componente qualquer na tela do computador e transmitir a informação por meio de interfaces de comunicação entre o computador e um sistema de fabricação, de modo que o componente pode ser produzido automaticamente numa máquina CNC (USINAGEM & AUTOMAÇÃO, 2009).

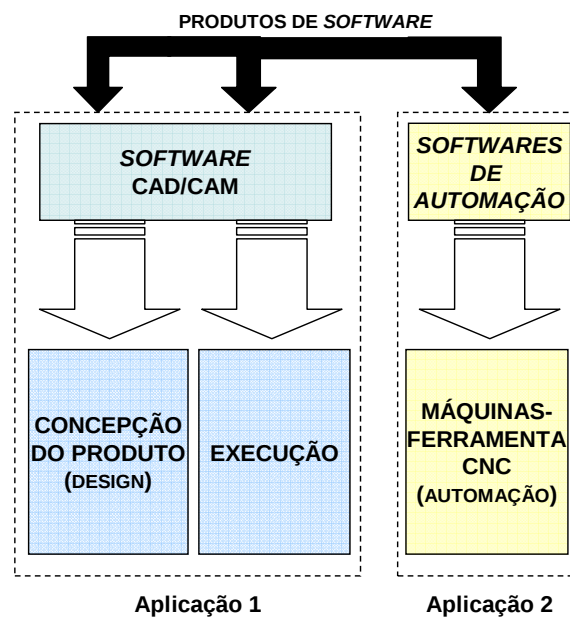


Figura 2 – Aplicações de produtos de *software* na indústria.

A segunda forma opera com o auxílio do *software* no controle da máquina, em que o homem é deslocado do centro decisório do processo produtivo, restringindo-se ao papel de supervisão. Paralelamente, evidenciam-se possibilidades proporcionadas pelo

⁸ Para Cunha (1987, p.36) “os progressos desse novo ramo da ciência foram lentos durante algum tempo, principalmente na década de 50, em virtude da inadequação dos computadores então existentes para uso interativo e também pela não disponibilidade de dispositivos gráficos”.

software, tal como demonstrado no estudo de Elenice, que tornam mais complexas as tarefas de preparação da máquina, passando a exigir novos conhecimentos e habilidades dos trabalhadores, devido à possibilidade do *software* ser reprogramado no chão de fábrica, permitindo ajustes conforme as necessidades de produção e variações do produto. Tais possibilidades de reprogramação devem-se à simplificação na mediação entre homem e máquina permitida pelas linguagens de programação.

Para uma devida apreensão do desenvolvimento das linguagens de programação, ou linguagens computacionais é necessária uma aproximação a partir do desenvolvimento do próprio computador (máquina) que vem a constituir a unidade de Comando Numérico Computadorizado. É importante verificar esse desenvolvimento desde os comandos mecânicos até os comandos proporcionados pela microeletrônica (item 2.1), para em seguida retomar mais especificamente alguns conceitos referentes às linguagens de *software* (item 2.2), que servirão, no capítulo terceiro, para o entendimento do processo de codificação do conhecimento.

ITEM 2.1

HISTÓRICO RESUMIDO DO DESENVOLVIMENTO DO *HARDWARE* E DO *SOFTWARE*

Desde a Antiguidade, as necessidades de automatizar processos levaram ao desenvolvimento de instrumentos mecânicos para calcular. Supõe-se que as primeiras tentativas de contagem foram feitas pelos pastores de ovelhas que marcavam a areia com sulcos ou empilhavam pedrinhas ou gravetos na medida em que as ovelhas passavam. O instrumento mecânico mais antigo de que se tem conhecimento para o auxílio à quantificação é o **ábaco**⁹.

Em 1614, John Napier, matemático escocês, criou os Bastões de Napier, que consistiam num conjunto de 9 bastões, os quais, ao serem ajustados uns aos outros permitiam multiplicar e dividir os números, simplificando essas operações para números grandes, ao reduzi-las a simples operações de adição e subtração. William Oughtred representou os logaritmos de Napier em escalas, que receberam o nome de “círculos de proporção”. O dispositivo de Oughtred originou a **régua de cálculos**. Outras três invenções marcaram o século XVII em relação ao desenvolvimento e aperfeiçoamento de máquinas mecânicas com finalidade de cálculo. Em 1623 e 1640, o alemão Wilhelm Schickard e o francês Blaise Pascal criaram, respectivamente, as primeiras calculadoras mecânicas com operações de soma e subtração com números de seis, no primeiro, e sete dígitos, no segundo caso. Em 1672, Leibniz, aperfeiçoa a calculadora de Blaise Pascal,

⁹ Formado por uma moldura de arames paralelos, dispostos no sentido vertical, correspondente cada um a uma posição digital referente às unidades, dezenas, centenas e milhares, tem os elementos de contagem constituídos por esferas com o centro perfurado, interceptado pelos arames paralelos sobre os quais deslizam. O ábaco pode ser considerado como uma extensão do ato natural de contar os dedos, que emprega um processo de cálculo com sistema decimal, atribuindo a cada haste um múltiplo de dez, permitindo dessa forma, a execução das quatro operações básicas da matemática. Com origem provável na Mesopotâmia há 2.000 anos, constitui um marco na referência histórica dos instrumentos mecânicos para auxiliar cálculos, pois através do seu mecanismo simplificado de contar, estabelece regras memorizáveis pelo usuário, resolvendo dessa forma os problemas envolvendo as operações aritméticas básicas (INTERNET / MICROINFORMÁTICA, 2008).

incorporando a esta as demais operações matemáticas, permitindo dessa forma a execução de adição, subtração, multiplicação e divisão em um mesmo instrumento.

Durante o século XIX, as máquinas de cálculo tiveram o seu funcionamento aperfeiçoado. A programação de comandos apresentou evolução significativa quando, já em 1801, Joseph Marie Jacquard, mecânico francês, sugeriu controlar teares, ou seja, automatizar a máquina de tear por meio de **cartões perfurados**. Os cartões forneceria os comandos necessários para a tecelagem de padrões complicados em tecidos. Em um grau bastante reduzido, a máquina criada por Jacquard possui alguns elementos teóricos que a assemelham às máquinas de Comando Numérico Computadorizado, tal como a capacidade de serem programadas através de uma linguagem pré-definida, no caso, os cartões perfurados. Entretanto, tal como uma máquina de automação rígida, a máquina de Jacquard possuía um alto grau de dependência do trabalho vivo, além de ser operada em sua totalidade através de movimentos mecânicos.

Na primeira metade do século XIX, o engenheiro e matemático inglês Charles Babbage apresentou um projeto que consistia em uma máquina que tinha como finalidade o cálculo de tábuas logarítmicas usadas em navegação. Preocupado com os erros contidos nas tabelas matemáticas de sua época, construiu um modelo para calcular tabelas de funções (logaritmos, funções trigonométricas etc.) sem a intervenção de operador humano, que chamou de **máquina diferencial**. Ao operador cabia somente iniciar a cadeia de operações e a seguir a máquina tomava seu curso de cálculos, preparando totalmente a tabela prevista. Essa máquina baseava-se no princípio de discos giratórios¹⁰ e era operada por uma simples manivela, ou seja, o operador definia os parâmetros de

¹⁰ Círculos concêntricos dispostos em um mesmo eixo, sendo cada um dos discos responsável por uma parte da operação.

cálculo a partir dos ajustes dos discos giratórios e girava a manivela até que os cálculos fossem realizados. A Figura 3 – Processo de cálculo da máquina diferencial, baseado em PIROPO (2006). mostra como se dá o processo de cálculo numa máquina diferencial.

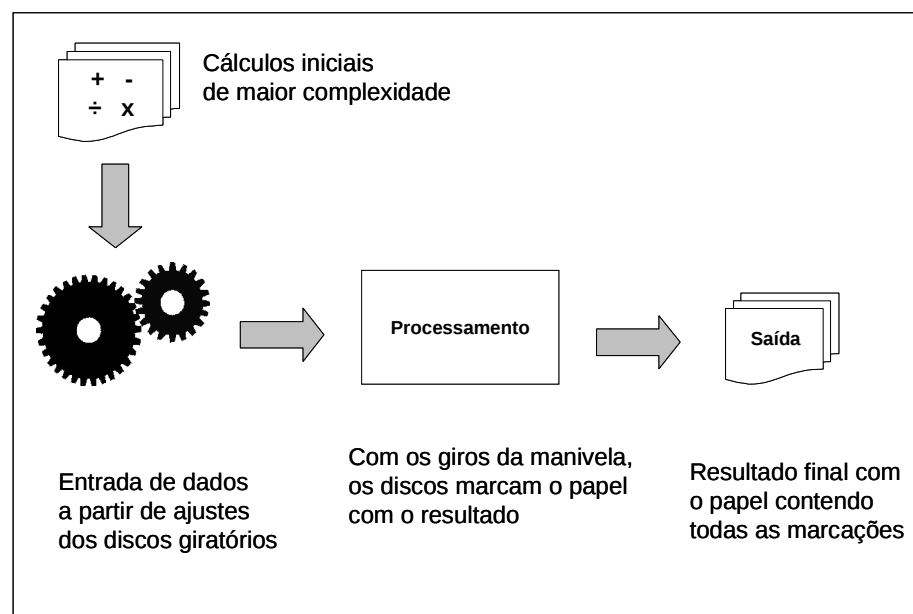


Figura 3 – Processo de cálculo da máquina diferencial, baseado em PIROPO (2006).

A máquina idealizada por Babbage tinha como finalidade o processamento da etapa de cálculo mais simples (no caso as diferenças) das equações para obtenção dos resultados das funções polinomiais. Conforme a Figura 3 – Processo de cálculo da máquina diferencial, baseado em PIROPO (2006)., a entrada de dados deveria ser realizada após a primeira etapa de cálculo, de maior complexidade, ser efetuada. Para essa primeira etapa seria necessário trabalho altamente qualificado, ou segundo Piropo (2006) “uma dúzia de matemáticos sábios”. Para a segunda etapa de cálculos de menor complexidade, seria necessário trabalho menos qualificado ou não especializado, capaz de realizar etapas contínuas e repetitivas de cálculo até que fosse atingido o resultado final. Com base na observação desse arranjo de trabalho, combinando trabalhadores altamente qualificados e menos qualificados, Babbage iniciou o desenvolvimento da máquina diferencial, influenciado

pela experiência anterior de De Prony, que Piropo (2006) descreve, comparando-a com o estudo de Smith sobre a fábrica de alfinetes¹¹:

[...] De Prony adotou um sistema similar [ao da fábrica de alfinetes], organizando sua “fábrica” de tabelas em três seções. A primeira era formada por uma dúzia de matemáticos e sábios (da qual fazia parte Lazare Carnot, o do princípio de Carnot sobre a eficiência das máquinas) que escolhiam as fórmulas e equações a serem usadas para calcular as tabelas. A segunda era formada por um grupo de supervisores, que organizavam o trabalho, compilavam os resultados, conferiam os cálculos e os preparavam para a impressão. E a terceira formada por quase uma centena de “computadores”, dos quais não se exigia grande conhecimento matemático mas apenas que soubessem executar as operações fundamentais com exatidão. A divisão do trabalho aplicada por De Prony para o cálculo de tabelas causou profunda impressão em Babbage. Mas ele sabia que uma das principais causas de erros nas tabelas não eram cálculos equivocados, mas a transcrição e impressão de resultados errados. Por isto concebeu uma forma de construção de tabelas que também usava uma distribuição de tarefas, como a implementada por De Prony, mas cuja parte mais trabalhosa, justamente aquela que exigia mais “força bruta” e menos conhecimento, o cálculo propriamente dito e sua impressão, fosse executada não por homens, mas por uma máquina. Assim nasceu a idéia da “*Difference Engine*”, a “máquina diferencial” de Babbage.

Babbage continuou a desenvolver e aperfeiçoar a máquina diferencial durante aproximadamente 10 anos, mas não chegou a concluí-la¹². Em 1833, focou seus esforços no desenvolvimento de uma nova máquina que, diferentemente da máquina diferencial, seria capaz de executar o cálculo de forma completa e não apenas uma parte do processo. Esse dispositivo seria a construção do que poderia ser chamado de computador mecânico-automático digital ou **máquina analítica**.

¹¹ Para entendermos o conceito é preciso recuar até o final do século dezoito, quando Smith publicou sua obra seminal, “*The wealth of Nations*” [A riqueza das nações], um tempo em que não havia fábricas e toda a produção de bens era feita artesanalmente, cada objeto sendo “fabricado” do princípio ao fim pelo mesmo artesão. No livro, Smith usa a manufatura de alfinetes como exemplo de uma nova concepção de fábrica baseada na divisão de tarefas. Segundo ele, para fabricar alfinetes era preciso cortar um fio de arame em pedaços do tamanho do alfinete, rebater uma das extremidades para criar a cabeça, afinar a outra para formar a ponta, polir o alfinete, embalar os alfinetes prontos e assim por diante. Smith comparou o resultado do trabalho de um único operário encarregado de todas estas operações com o obtido a partir da divisão de tarefas por uma equipe na qual cada operário seria responsável por uma e apenas uma das ações, demonstrando que esta divisão resultaria em uma especialização que por sua vez redundaria em um enorme ganho de produção (PIROPO, 2006).

¹² Em 1991, o *Science Museum* em Londres construiu a máquina diferencial de Babbage nº 2 para uma exposição sobre a história da computação (INTERNET / MICROINFORMÁTICA, 2008).

A máquina analítica era constituída de unidade de controle de memória, aritmética, de entrada e de saída de dados, no sentido da informação processada pela máquina. Sua operação era comandada por um conjunto de cartões perfurados, de modo que, de acordo com os resultados intermediários, a máquina poderia saltar os cartões, modificando dessa forma o curso dos cálculos (FERRARI, 2008). A transferência, dessa forma, das tomadas de decisões para a máquina decorre das programações realizadas inicialmente pelos programadores. Esta última característica é de extrema importância uma vez que o controle do processo se transfere para a concepção, ou seja, para a etapa de programação da máquina, ou seja, ao invés de “sábios matemáticos” capazes de executar os cálculos iniciais, seriam necessários programadores capazes de “preparar” os cartões perfurados. O conceito da máquina analítica de Babbage pode ser descrito a partir de um modelo simplificado constituído de entrada, processamento e saída de dados (Figura 4).

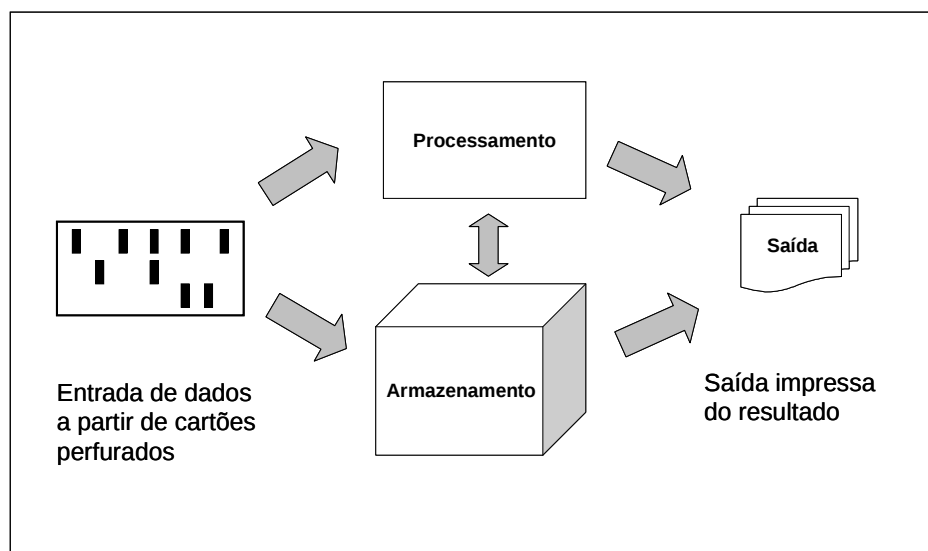


Figura 4 – Modelo de entrada, processamento e saída de dados da máquina analítica de Babbage baseado em MARTIN, M.B & LOCH, M. (1999).

A entrada de dados é pré-concebida e programada pelo programador, o processamento é realizado – de forma mecânica – pela máquina, a qual pode armazenar

mil números de até cinquenta dígitos. O resultado impresso é dado a partir da marcação em cartões pela própria máquina.

A matemática inglesa Ada Augusta King, Condessa de Lovelace estudou a proposta da máquina diferencial de Babbage e criou um programa para calcular certa seqüência de números. Embora nunca tenha sido executada de fato, a programação realizada para a máquina de Babbage foi descrita por Lovelace, conforme Brooks (1986, p. 68-70) como “tecendo modelos algébricos do mesmo modo que o tear de Jacquard tecia flores e folhas”.

Caso a **máquina analítica** houvesse sido completada, teria implementado o modelo de entrada (cartões perfurados com a seqüência inicial de instruções preparadas pelo operador da máquina), processamento-armazenamento (execução do processo de cálculo efetuado pela máquina) e saída (cartões perfurados com o resultado do cálculo efetuado pela máquina) que constitui a arquitetura básica da computação. A programação da **máquina analítica** de Babbage, realizada por Ada Lovelace, estabeleceu as estruturas básicas para a programação de condições pré-estabelecidas para a tomada de decisões de forma automatizada. Lovelace estudou o modelo de entrada-processamento-saída proposto por Babbage na máquina analítica e elaborou programas que, no âmbito teórico, poderiam ser executados nas máquinas que implementassem tal modelo (MARTIN, M.B & KOCH, M., 1999). Lovelace sugeriu algumas seqüências de instruções, repetições e decisões condicionais a serem processadas pela máquina que mais tarde viriam a ser visualizadas como rotinas de programação. A Figura 5 mostra a estrutura estabelecida.

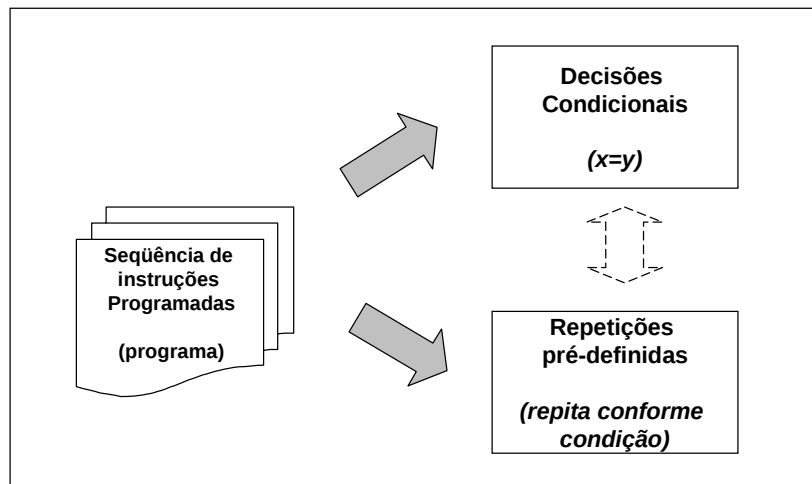


Figura 5 – Estruturas básicas para a programação baseado em GONICK (1986).

Conforme o programa criado por Ada Lovelace, a sequência de instruções de entrada deve obviamente seguir um padrão pré-determinado que seja compreendido pela máquina, ou seja, embora aparentemente aleatórios, os orifícios dos cartões perfurados devem obedecer a padrões de dimensão, tamanho e espaçamento entre os furos. As decisões condicionais possibilitam que, a partir de determinadas condições pré-estabelecidas no programa, a máquina possa assumir decisão condicionada à sequência lógica pré-definida. A máquina nada decide, já que todas as condições possíveis de serem tomadas estão pré-determinadas pelo programador conforme as condições definidas no programa de entrada. Lovelace pode ser considerada a primeira programadora da história, projetando e explicando, a pedido de Babbage, programas para a máquina idealizada. Lovelace cunhou desde os conceitos de sub-rotina, uma sequência de instruções que pode ser usada várias vezes, *loop*, uma instrução que permite a repetição de uma sequência de cartões, e de salto condicional, o ato de saltar algum cartão caso uma condição seja satisfeita (MENEZES, 2007).

As repetições, aliadas às decisões condicionais, permitem à máquina, a repetição de ininterruptas rotinas pré-estabelecidas pelo programador. Considerando o conceito de *hardware* como sendo a parte física do computador, basicamente constituída de circuitos integrados e o conceito de *software* como um conjunto de produtos desenvolvidos para serem operados no *hardware*, historicamente Babbage pode ser considerado o precursor do computador e Ada Lovelace, a precursora da programação¹³.

Entre 1850 e 1900 surgiram grandes avanços na matemática e na física, que se traduziram no desenvolvimento das equações diferenciais. O desenvolvimento e aperfeiçoamento da máquina a vapor permitiu o avanço da manufatura e transporte em um período marcado por intensos avanços na engenharia. Os projetos dos trilhos de trens, navios a vapor, moinhos têxteis e pontes pressupunham o conhecimento de cálculo estrutural para a determinação de importantes informações, tais como: centro de gravidade, centro de flutuabilidade ou cálculo da linha d'água, utilizado em projetos de embarcações, momento de inércia e distribuição de carga. Mesmo a avaliação da força de saída dos motores a vapor necessita de integração matemática para ser calculada, havendo, portanto, uma grande necessidade de máquinas que pudessem efetuar cálculos de forma repetitiva, precisa e rápida. Para Marx (1980, p. 441), a ciência que possibilitava mensurar fenômenos da natureza permitiu à indústria moderna, progressivamente,

[...] apoderar-se de seu instrumento característico de produção, a própria máquina, e de produzir máquinas com máquinas. Só assim criou ela sua base técnica adequada e ergueu-se sobre seus próprios pés. Com a produção mecanizada crescente das

¹³ Em uma comparação entre a máquina analítica de Babbage e a máquina de Controle Numérico Computadorizado, composta esta por programador, linguagem computacional, unidade de controle, máquina e operador, a máquina de Babbage corresponde aos itens linguagem computacional (cartões perfurados), unidade de controle e máquina, tendo como insumo os cartões perfurados e como saída o resultado dos cálculos processados. Tal comparação é limitada, pois a máquina analítica de Babbage tem como finalidade o processamento de cálculo (produz cartões perfurados com o resultado das informações inseridas na máquina pelo programador) e a máquina de Controle Numérico Computadorizado é responsável pelo controle de outras máquinas em um processo de produção industrial.

primeiras décadas do século XIX, apoderou-se a maquinaria progressivamente da fabricação das máquinas-ferramenta. Mas só durante as últimas décadas (que precederam 1866), a enorme construção de ferrovias e a navegação transatlântica fizeram surgir as máquinas ciclópicas empregadas na construção de motores.

Com a exploração comercial da eletricidade, a partir de 1880, quando Thomas Edison patenteia o sistema de distribuição elétrica, um ano após ter inventado a lâmpada, o paradigma das máquinas com o propósito de processamento com funcionamento mecânico é quebrado, alterando o curso de inovações e aperfeiçoamento de conceitos existentes (MENEZES, 2007). Até então as máquinas de processamento estavam sujeitas a um funcionamento totalmente mecânico, seja por motores a vapor ou por força humana, utilizando alavancas, manivelas e pedais.

Um passo importante para a criação dos computadores da era moderna foi a utilização e aperfeiçoamento dos cartões perfurados, os quais foram utilizados com sucesso em 1890 por Herman Hollerith e James Powers que trabalhavam para o censo dos Estados Unidos. Hollerith e Powers desenvolveram dispositivos capazes de ler informações a partir de cartões perfurados sem a interferência humana. Este fato reduziu erros de maneira acentuada, além de reduzir igualmente a quantidade de pessoas envolvidas no trabalho. A máquina desenvolvida por Hollerith foi pioneira em utilizar a eletricidade na separação, contagem e tabulação dos cartões. Os dados do censo de 1880, manualmente processados, levaram 7 anos e meio para serem compilados. Com o uso da máquina, o censo de 1890 foi processado em 2 anos e meio. As informações eram armazenadas por meio de perfurações em locais específicos do cartão. Nas máquinas de tabular, um pino passava pelo furo e chegava a uma jarra de mercúrio, fechando um circuito elétrico e causando um incremento de 1 em um contador mecânico. Um outro fator importante é que os cartões poderiam ser classificados e armazenados conforme os

dados provenientes do censo, podendo ser acessados eventualmente e reavaliados conforme a necessidade.

O matemático britânico George Boole descreveu, em seu livro “Uma Investigação das Leis do Pensamento”, um sistema para o raciocínio lógico e simbólico que permitiu posteriormente que as informações utilizadas e manipuladas em computadores pudessem ser armazenadas e processadas utilizando relações binárias. Segundo o próprio Boole (1854: p. 1),

o motivo do presente tratado é investigar as leis fundamentais do funcionamento do cérebro através das quais o raciocínio se realiza; expressá-las através da linguagem do cálculo e, sobre este fundamento, estruturar a ciência da lógica e construir o seu método; fazer deste método a base de todos os métodos para aplicação da doutrina matemática das probabilidades; e, finalmente, recolher dos vários elementos verdadeiros trazidos para serem examinados no curso destas investigações alguma provável sugestão a respeito da natureza e constituição da mente humana.

É possível efetuar um paralelo entre o funcionamento da idealizada máquina diferencial de Babbage com a lógica definida por Boole e o funcionamento dos cartões perfurados. Na máquina diferencial, após os cálculos iniciais de maior complexidade, era necessário o ajuste dos discos giratórios para que os cálculos de menor complexidade fossem executados. Boole utiliza os cartões perfurados como entrada de informações, tal como os cartões perfurados utilizados no tear mecânico proveniente da invenção de Joseph Marie Jacquard. Os cartões perfurados se encaixam na lógica de Boole perfeitamente, pois os orifícios dos cartões indicam os uns e áreas não perfuradas indicam os zeros. O sistema de cartões perfurados mostrou que as máquinas poderiam ser controladas pelo sistema binário de maneira extremamente satisfatória (BONATTI & MADUREIRA, 1990).

Tais características foram gradativamente aperfeiçoadas por Hollerith e também utilizadas por outras empresas com finalidades comerciais. Em 1924, cinco anos antes da morte de Herman Hollerith, foi fundada a International Business Machines Corporation, ou IBM. A empresa, criada a partir da idéia de máquinas automatizadas que utilizavam cartões perfurados tanto como entrada de dados quanto como saída de informações processadas, tinha a proposta inovadora de desenvolvimento de máquinas com processamento de natureza genérica. Diferentemente das máquinas com funcionamento puramente mecânico, a partir do uso comercial e difusão da eletricidade, as máquinas que faziam uso de cartões perfurados, similares às criadas por Hollerith, utilizavam a eletricidade para movimentar mecanicamente as peças, alimentar os cartões perfurados de entrada, adicionar, multiplicar, escolher, no sentido de adotarem decisões de forma condicionada, e alimentar os cartões perfurados com os dados processados de saída.

Em 1937, com a tese de mestrado de Claude E. Shannon “Uma análise simbólica de relê e circuitos de chaveamento” foi possível, através dos conceitos da álgebra de Boole, a sua aplicação em conjuntos eletromecânicos para resolver problemas relativos à lógica. Tal idéia, que utiliza propriedades de circuitos eletrônicos para a lógica, é o conceito básico que mais tarde culminaria nos computadores digitais¹⁴. Claude Shannon

¹⁴ No seu relevante trabalho *Uma Teoria Matemática da Comunicação*, publicado em 1948 na revista *Bell System Technical Journal*, Shannon desvenda uma área inédita na matemática, que surgiu essencialmente com as definições e conceitos básicos formulados, com os teoremas e resultados fundamentais já estabelecidos e sem precedentes visíveis na literatura existente na época. Nesse trabalho Shannon desenvolve a teoria da informação e transmissão de sinais digitais, baseados em sequências de zeros e uns. É aí que define "o problema fundamental da comunicação como o de reproduzir num local, de forma aproximada ou exata, uma mensagem selecionada noutro local" (SHANNON, 1948, p.379). Estabeleceu então o esquema de transmissão de informação hoje clássico, com uma mensagem que parte de uma fonte, é codificada e emitida por um transmissor, passa por um canal de comunicação, sofre perturbações designadas por ruídos e chega depois ao receptor, passando por um sistema de decodificação. Ao falar de "uma mensagem selecionada", Shannon refere-se a uma sequência informativa que pode ser escolhida entre muitas outras, que aparecerão com iguais ou diferentes probabilidades. Define então a quantidade de informação com base na sua incerteza ou dificuldade de previsão (SHANNON, 1948). A Teoria da Informação, desenvolvida a partir da década de 1940, é um dos pilares da chamada Era Digital. A compreensão de seus conceitos favoreceu o desenvolvimento de aplicações jamais pensadas até então, elevando ao nível de ciência a atividade de engenheiros e técnicos daquela época. Ainda, demonstrou

desenvolveria, no seu célebre artigo de 1948, intitulado “A teoria matemática da comunicação”, a teoria da informação que serve de fundamento para áreas de estudo como compressão de dados e criptografia, que seriam utilizados amplamente pelos computadores desenvolvidos e utilizados na Segunda Guerra Mundial.

O primeiro **computador eletromecânico** foi criado em 1936 pelo Alemão Konrad Zuze. A máquina construída com a finalidade de cálculo utilizava o sistema binário e podia ser programada de forma simples e mecânica a partir do remanejamento de relés eletromagnéticos, dispositivo utilizado para representar os estados de ligado ou desligado, implementando dessa forma a lógica booleana de forma simplificada. Os componentes capazes de implementar a lógica interna de cálculo com base no sistema binário foram fundamentais para o aperfeiçoamento e desenvolvimento dos computadores ao longo da história e principalmente durante a Segunda Guerra Mundial. Tais componentes podem ser resumidamente descritos como:

- Relé eletromagnético (1835): inventado pelo Professor de física Joseph Henry da Universidade de Princeton, foi utilizado para aperfeiçoar o sistema de comunicação criado por Samuel Morse e um século mais tarde utilizado na primeira calculadora eletromecânica criada por Konrad Zuze.
- Válvula (1880): criada por John Ambrose Fleming, utilizada em experimentos de radiofrequência e radiodifusão e 10000 vezes mais rápida que os relés eletromagnéticos, foi utilizada pelo exército britânico em 1943 na construção da máquina de decodificação de mensagens.

- Transístor (1947): criado nos laboratórios da empresa estadunidense de telefones e telégrafos American Telephone & Telegraph (AT&T) em dezembro de 1947 por Bardeen e Brattain, tinha como objetivo criar um dispositivo compacto e barato para substituir as válvulas utilizadas nos sistemas telefônicos da época.
- Circuito Integrado (1958): criado por Jack S. Kilby, da empresa Texas Instruments, e Robert N. Noyce, da empresa Fairchild Semiconductor, consistia na idéia de juntar um único circuito com vários transistores, uma única peça impressa sobre uma placa de semicondutor de silício. Kilby pensava em maneiras de miniaturizar os componentes e simplificar sua fabricação, possivelmente construindo todos sobre uma mesma lâmina de silício.
- Microchip (1970): em um projeto coordenado pelo físico italiano Federico Faggin (1968), os cientistas Andy Grove, Robert Noyce (co-inventor do circuito integrado) e Gordon Moore, oriundos da Fairchild Semiconductor, fundaram a empresa Intel em 1968, desenvolvendo posteriormente o que seria um complexo circuito com centenas de milhares de transístores, capaz de realizar todas as tarefas lógicas e matemáticas do processamento, num único *chip*.

O início da Segunda Guerra Mundial acarretou a necessidade de computadores capazes de realizar cálculos cada vez mais precisos e rápidos, intensificando o investimento em pesquisa e desenvolvimento de máquinas computadorizadas para explorar o seu potencial estratégico militar, assim como auxiliar nos cálculos de projetos de novos aviões, mísseis, tabelas de trajetória de alvos e cálculos balísticos. Em 1943, o exército inglês desenvolveu um computador, denominado *Colossus*, capaz de decodificar mensagens alemãs, porém com impacto comercial bastante inexpressivo devido à especificidade da máquina e o sigilo do projeto.

A pesquisa e desenvolvimento norte americanos produziram efeito mais contundente a partir do investimento da marinha no projeto de Howard H. Aiken, doutor em física pela Universidade de Harvard, que trabalhava para a IBM, para a criação do Mark I, dispositivo eletro-mecânico de computação capaz de automatizar o cálculo de equações diferenciais, em 1944. O trabalho de Aiken possuía influência da máquina analítica de Babbage e resultou em uma máquina extremamente mais complexa que aquela desenvolvida por Konrad Zuse. Simultaneamente, e em segredo, o exército norte americano desenvolvia um projeto semelhante, chefiado pelos engenheiros J. Presper Eckert e John Mauchly, cujo resultado foi o primeiro computador a válvulas, diferente do Mark I, constituído por relés. O Electronic Numeric Integrator and Calculator – ENIAC seria capaz de fazer quinhentas multiplicações por segundo, tendo sido projetado para calcular trajetórias balísticas. O ENIAC foi o primeiro computador totalmente programável eletronicamente. Conseguiu fazer 500 multiplicações por segundo, mas possuía custos para manutenção e conservação inviáveis sob o ponto de vista comercial¹⁵. Dezenas de válvulas queimavam a cada hora e o calor gerado necessitava ser controlado por um complexo sistema de refrigeração. É importante verificar que o ENIAC utilizava a mesma arquitetura proposta pela máquina de Babbage – entrada de dados, processamento com armazenamento e saída de informações processadas – e os conceitos de programação de Ada Lovelace para execução das rotinas e decisões condicionais.

O matemático húngaro John Von Neumann formalizou então o projeto lógico do computador, sugerindo que as instruções fossem armazenadas na memória do mesmo, ao

¹⁵ Com 17.468 válvulas termoiônicas, 70.000 resistências, 10.000 capacitores e 1.500 relés o ENIAC pesava 28 toneladas, ocupava uma área de 72 m² na Universidade da Pensilvânia, E.U.A., de 270m², e queimava cerca de 150 kilowatts de energia (UFPA, 2009).

invés de serem lidas a partir de cartões perfurados e executadas em seguida. A idéia proposta por Neumann ampliava a aplicabilidade das máquinas de cálculo eletrônico para processadores de informações genéricas. Nesse sentido, a arquitetura proposta por Von Neumann tinha três características básicas: instruções codificadas de uma forma possível de ser armazenada na memória do computador de forma binária, aumentando assim a agilidade na leitura e processamento da informação; instruções armazenadas na memória, bem como toda a informação necessária à execução da rotina pré-estabelecida; e processamento do programa extraindo as instruções diretamente da memória do computador, ao invés de ser lido um novo cartão perfurado a cada tarefa a ser executada. Essa arquitetura define um **computador seqüencial** em que o processamento das informações é feito passo a passo, caracterizando um comportamento determinístico, ou seja, os mesmos dados de entrada produzem sempre a mesma resposta.

Von Neumann projetou o computador automático variável eletrônico EDVAC em 1945, com uma memória capaz de manter um programa armazenado, bem como os relativos dados. A memória de armazenamento e os controles condicionais permitiam que o computador fosse parado em qualquer ponto e retomasse então de novo o seu processamento, permitindo assim uma maior versatilidade na sua programação. Em 1951, o UNIVAC I, computador automático universal construído pela empresa Remington Rand, uma companhia norte americana fabricante de equipamentos para escritórios, foi o primeiro computador de válvula produzido em série e vendido comercialmente.

Segundo Pacciti (1998), o período compreendido entre 1945 e 1959 constitui a **primeira geração** de computadores, os quais faziam uso de válvulas eletrônicas, fios e possuíam grandes dimensões (o UNIVAC I ocupava cerca de 35 m²). Tinham como

característica, instruções programadas de forma limitada conforme a tarefa específica para a qual fossem utilizados. Cada computador tinha um programa código binário diferente chamado de linguagem de máquina que descrevia como a máquina deveria operar. A linguagem computacional complexa dos computadores da primeira geração, aliada à quantidade de válvulas necessárias à realização das operações tornava o computador difícil de programar e limitou a sua versatilidade e velocidade. Baseado no uso de válvulas e com custo de manutenção e periféricos bastante elevados¹⁶, os computadores da primeira geração tiveram uso voltado para grandes instituições que pudessem arcar com os elevados custos, tal como universidades, órgãos governamentais e instituições bancárias. A utilização das máquinas estava voltada para trabalhos administrativos e científicos programados em linguagem de máquina. Os trabalhos eram programados na máquina por intermédio de cartões perfurados, com elevado grau de complexidade, e uma vez iniciados não poderiam ser interrompidos. Durante a primeira geração de computadores iniciou-se um intenso trabalho de pesquisa e desenvolvimento da empresa IBM, mesclando na equipe consultores renomados como matemático Von Neumann e engenheiros recém-formados, o que culminou com a produção de máquinas científicas e comerciais como o IBM 701 e IBM 702, que possuem uma versão inicial da linguagem Fortran (PACITTI, 1988).

Em 1958, a invenção do transistor alterou o curso do desenvolvimento do computador conduzindo à **segunda geração** de máquinas, de maior confiança de cálculo do que suas predecessoras. Essa segunda geração, compreendida entre 1959 e 1964 (PACITTI, 1998), além de substituir as válvulas eletrônicas por transístores, substituiu os fios de ligação dos computadores da geração anterior por circuitos impressos, o que os

¹⁶ No fim da década de 50, uma impressora *off-line* do Univac-1105 custava cerca de meio milhão de dólares (PACITTI, 1988).

tornou mais rápidos. Os cálculos, anteriormente medidos em segundos, passaram a ser medidos em microssegundos e, conseqüentemente, com custos de produção mais baixos.

A **terceira geração** inicia-se, segundo o autor, a partir de 1965, com o uso de circuitos integrados, proporcionando maior compactação, redução dos custos e aumento da velocidade de processamento. Os circuitos integrados (CI) miniaturizam centenas e milhares de transístores em uma pastilha de silício (*chip*).

No início da terceira geração ocorre o sucesso do IBM 360¹⁷ que iniciou importantes avanços na padronização do *hardware* e influenciou consideravelmente o desenvolvimento de *softwares*. Além de estabelecer padrões de organização de memória interna (*hardware*), o projeto da nova máquina definiu inovações nas unidades de fita, impressora e leitor de cartões. Além disso, o IBM 360 passou a adotar o caractere como a menor unidade de acesso à memória, em detrimento de palavras de maior tamanho¹⁸, que dentre outras dificuldades para os usuários, ocasionava uso desnecessário de memória. Essa mudança abriu as possibilidades e ampliou as aplicações dos computadores, pois a partir de então, outros fabricantes começaram a adotar toda a padronização utilizada no IBM 360, propiciando a comunicação entre sistemas e referência de padrões. Graças à padronização estabelecida pelo modelo 360, os programas de *software* desenvolvidos poderiam funcionar adequadamente nos modelos seguintes, como aconteceu com o modelo 370, comercializado na década de 70. Devido aos excessivos custos de

¹⁷ Seis meses após ser anunciado o modelo 360 ao mercado, suas encomendas ultrapassaram três vezes a receita anual anterior da empresa IBM. Devido ao fato, alguns competidores saíram do mercado (PACITTI, 1988).

¹⁸ A adoção do caractere como menor unidade de acesso à memória consolidou o código ABCDIC (*Extended Binary Coded Decimal Interchange Code*) em que cada caractere (alfabético, algarismo, pontuação e sinais de controle) tem uma única representação em um conjunto de oito *bits*, também chamado octeto ou *byte*. Com oito bits pode-se representar 256 caracteres. O código ABCDIC antecipou-se ao novo padrão de transferência de caracteres no campo das telecomunicações, chamado padrão ASCII (*American Standart Code for Information Interchange*), que usa sete dígitos binários, ao qual o ABCDIC facilmente se adaptou (PACITTI, 1988).

programação incorridos nos modelos anteriores dos diversos fabricantes, a compatibilidade e reusabilidade de *softwares* a partir do modelo 360 encorajaram a aquisição de modelos da IBM, assim como o desenvolvimento de *software* orientado pelos novos padrões. Dessa forma, a portabilidade de programas de *software* em máquinas de um mesmo fabricante evoluiu para outras máquinas de diferentes fabricantes, na medida em que aderiam à padronização estabelecida a partir de então.

Devido à padronização iniciada pelos modelos da empresa IBM, o final da década de 60 pode ser considerado como a década do surgimento da indústria do *software*, quando surgem as primeiras *software houses* (PACITTI, 1988). Até o final da década de 60 os *softwares* eram fornecidos pelos próprios fabricantes de *hardware* e só funcionavam para a máquina específica juntamente com o próprio **sistema operacional** do fabricante da máquina. As *software houses* se apoiavam no conceito de portabilidade de *software* com o intuito de um mesmo *software* poder funcionar em qualquer máquina (*hardware*), independente do fabricante.

As *software houses* desenvolveram-se rapidamente no nicho de mercado de computadores de médio porte¹⁹, gerando novos protocolos, **sistemas operacionais**, **linguagens de programação** e bancos de dados. Nesse sentido, as *software houses* puderam competir em um mercado antes dominado pelos fabricantes de *hardware* (*mainframe*). Durante a década de 60, foram introduzidos no mercado, os chamados

¹⁹ Historicamente, os computadores podem ser classificados por categoria como grande porte, *mainframes*, médio porte, minicomputadores e pequeno porte. Conceitualmente, independentemente do porte, todos realizam funções internas idênticas, porém em escalas diferentes. Os *mainframes* se destacam por terem alto poder de processamento e muita capacidade de memória e controlam atividades com grande volume de dados, sendo de custo bastante elevado. Os *mainframes*, ou computadores de grande porte, dominaram a primeira e a segunda geração de computadores e começaram a dividir mercado com os minicomputadores (médio porte) a partir do início da terceira geração. A miniaturização gradual dos *mainframes* para os minicomputadores e microcomputadores é decorrente do desenvolvimento e aperfeiçoamento tecnológico gradual ocorrido desde o relé eletromagnético, válvula, transistor, circuito integrado e *microchip* (HENNESSY & PATTERSON, 2005).

minicomputadores, com tecnologia de pequeno porte, permitindo que a máquina fosse dedicada a um único usuário, ao contrário dos *mainframes* que se caracterizavam pela convivência de várias estações de trabalho ligadas a uma única central de processamento. Conforme Pacitti (1988), a utilização dos minicomputadores foi rapidamente disseminada em diversos setores, pois

Os minicomputadores, de uma maneira geral, tinham maiores aplicações no controle de processos, nas atividades acadêmicas, e menos na grande área administrativa. Podiam ser colocados em pequenos laboratórios, salas de estudo; a marinha [norte] americana os colocava a bordo de suas embarcações. [...] Os minicomputadores foram a primeira ameaça aos grandes computadores centralizados, os *mainframes*.

Paralelo ao avanço do *hardware*, as linguagens da terceira geração se popularizaram em decorrência de proporcionarem uma programação de alto nível ou de baixa complexidade para o usuário. Os computadores dessa geração passaram a ser mais amigáveis (FERRARI, 2009). Se, por um lado, o *hardware* progredia no sentido da miniaturização dos componentes, o *software* direcionava os esforços no sentido da estruturação e portabilidade dos sistemas de programação, de modo que estes funcionassem em diversas plataformas independentemente do fabricante. O início da produção em massa do *hardware* permitiu o barateamento do mesmo assim como a sua intensa utilização.

Com a terceira geração de computadores tem início a utilização sistemas operacionais mais organizados, que são *softwares* que constituem o sistema central de controle e coordenação dos demais programas, bem como a gestão da memória do próprio computador. Segundo Gasparetti (2001), a partir de 1970 com o surgimento do microprocessador, teria início a **quarta geração** de computadores caracterizada pelo aperfeiçoamento da tecnologia consolidada na terceira, proporcionando uma otimização

da máquina para as respectivas finalidades de uso²⁰, maior grau de miniaturização das peças e componentes, confiabilidade e maior velocidade de execução das instruções, passando para a ordem de nanossegundos, bilionésima fração do segundo.

Em compasso com o desenvolvimento do *hardware* e do *software*, intensifica-se no Japão um movimento iniciado a partir da Segunda Guerra Mundial, que se consolidou sobretudo nas fábricas montadoras de automóveis a partir da década de 70, caracterizado pela flexibilidade e rapidez na montagem, valendo-se do sistema *just in time* que reduz ao mínimo os estoques. No seu impulso de conquistar mercados estrangeiros, as indústrias japonesas aplicaram amplamente tecnologias microeletrônicas, a fim de aumentar a produtividade e melhorar a qualidade dos seus produtos. O toyotismo, como parâmetro das novas formas internacionalizadas de controle do trabalho, supõe uma intensificação na exploração, reduzindo ainda mais que o fordismo, as porosidades e aumentando o ritmo do processo produtivo (Antunes, 1995)²¹. Conforme Rattner (1988), a introdução e difusão da tecnologia microeletrônica no Japão têm início ao redor de 1975, avançando consideravelmente, sobretudo no setor manufatureiro com o posterior uso da robótica na automação.

²⁰ A empresa HP que tem no nome a sigla de seus fundadores, Hewlett-Packard, entrou no mercado de computadores em 1966, com o modelo HP2116, o primeiro de uma série usada nas aplicações CIM (*Computer-Integrated-Manufacturing*), que integram as funções de contabilidade de escritório (automação de escritório) com o sistema de automatizado de produção. Em 1972, a HP introduziu na área comercial a série 3000 (médio porte), um sistema de múltiplos usuários com grande confiabilidade para seu tempo. Também em 1972 introduziu a primeira calculadora científica manual, o modelo HP35, que tornou obsoleta, de vez, a antiga régua de cálculos dos engenheiros. Mais tarde a empresa entrou no mercado de microcomputadores (PACITTI, 1999).

²¹ Podemos caracterizar o fordismo como produção em massa rígida alicerçada no trabalho vivo, e o ohnoísmo como produção em massa flexível igualmente alicerçada no trabalho vivo. Este fato crucial fornece ao fordismo/ohnoísmo sua diferença específica relativamente à produção em massa lastreada na maquinaria, caso típico das indústrias têxtil e de processo contínuo. Ora, a automação de base microeletrônica terá como consequência permitir às indústrias de cunho fordista ou ohnoísta passar a alicerçar a produção em massa (necessariamente flexível) na maquinaria, e não mais no trabalho vivo. Isto significará, simplesmente, o fim histórico do fordismo, e de sua “reinvenção”, o ohnoísmo, e a emergência de um conceito unificado de produção industrial, que se constituirá, em todos os seus segmentos, numa “aplicação tecnológica da ciência” (MORAES NETO, 1998, p. 12).

Assim, mais do que pelo aperfeiçoamento do *software*, a quarta geração de computadores se caracterizou pela introdução da idéia de uma separação (e de uma relação) *hardware/software* nos processos produtivos. Já no início da quarta geração, por outro lado, o Departamento de Defesa norte americano – DOD, em um momento delicado da Guerra Fria e prevendo um possível ataque inimigo que eventualmente pudesse desarticular a rede de comunicação instalada, decide desenvolver uma rede de comunicação altamente descentralizada chamada Arpanet – *Advanced Research Project Agency*. Conforme Pacitti (1988, p. 177),

O Arpanet inicialmente ligou, através das novas tecnologias de redes, um grande número de universidades que faziam pesquisas científicas com fundos militares. Caso um dos “nós” desta rede fosse destruído pelo inimigo, o restante continuaria a funcionar normalmente. Razões altamente estratégicas. Logo de início, a Arpanet fez sucesso. Todas as universidades, outras organizações e suas redes desejavam interconectar-se com a Arpanet. Foi então que surgiram os problemas de gerência e controle das redes.

Mesmo com a separação das redes entre a Milnet (*Militar Net*) que interligava os *sites* militares e a NSF-NET (*National Science Foundation*) que agrupava os sites não-militares, a interligação entre ambas foi garantida graças a dois protocolos, o IP (*Internet Protocol*), que possibilita o tráfego de informações de uma rede para outra e o TCP (*Transmission Control Protocol*), que controla a transferência de dados (PACITTI, 1988). A dupla TCP-IP garantiu posteriormente o funcionamento da maior rede de computadores mundial, a *World Wide Web* popularmente conhecida como *Internet*. Outra característica fundamental dos protocolos é que permitem o funcionamento, interligação e comunicação de máquinas de diferentes plataformas e fabricantes na rede, bastando apenas a esses a implementação correta do protocolo. Dessa forma, os diversos computadores da rede

puderam se comunicar devidamente através do protocolo TCP-IP tal como aparelhos telefônicos de diferentes fabricantes se comunicam entre si através das redes telefônicas²².

Com o aperfeiçoamento do fax, que por sua vez foi um aperfeiçoamento e melhoramento no tempo de impressão elétrica e sinais telegráficos, foi possível produzir comercialmente o aparelho que, aliado aos microcomputadores, revolucionou as comunicações de dados à distância, o *modem*²³. Dessa forma, a partir do *modem* os usuários puderam conectar os seus computadores a qualquer computador distante utilizando a linha telefônica, barateando significativamente o custo da operação. Do ponto de vista da produção, coletivos de trabalho puderam se organizar em rede de forma virtual.

²² Conforme cita Bolaño (2004a, p.2), as comunicações em geral adquirem um papel central no novo padrão de acumulação que se desenha, pois o “desenvolvimento da internet, na verdade, está inserido nas grandes transformações por que passaram os diferentes setores das comunicações desde a reforma da radiodifusão européia dos anos 80 e a reestruturação global das telecomunicações dos 80 e 90, ligados à passagem a um novo modelo de regulação, mais adequado à nova estrutura do capitalismo, gestada ao longo da crise do padrão de acumulação do pós Segunda Guerra Mundial”.

²³ O Modem (Modulator Demodulator) converte dados digitais, provenientes da saída do computador, em sinais analógicos (tons de telefone), para serem transportados, em linhas telefônicas comuns, até a entrada de outro computador, onde esse sinal é restabelecido em sinal digital novamente, idêntico ao da origem (TANENBAUM, 2001).

ITEM 2.2

AS LINGUAGENS DE SOFTWARE

Uma linguagem de programação pode ser considerada como sendo uma notação que pode ser usada para especificar algoritmos com precisão. O **anexo I** apresenta todas as linguagens de programação por período, compreendidas entre 1954 e 2008. As linguagens de programação são utilizadas para descrever algoritmos ou seqüências de passos a partir de um conjunto finito de símbolos com os quais se escrevem programas de computador. Estabelecem operações básicas de computação, além de permitir que os programadores especifiquem como esses passos serão seqüenciados no programa de *software*.

Os computadores da primeira geração podiam ser programados em **linguagem de máquina** a partir dos cartões perfurados ou através da combinação de fios e cabos que alteravam o seu funcionamento. A segunda geração de computadores trouxe consigo linguagens mais acessíveis aos programadores, ou seja, menos detalhistas, descartando assim a necessidade de conhecimento específico e avançado em computação para a programação das máquinas. Para Pacitti (1988, pg. 114),

Os computadores de segunda geração, agora com circuitos de chaveamento transistorizados discretos, trouxeram consigo as memórias internas de núcleos magnéticos, com tamanho reduzido e grande confiabilidade (em relação aos anteriores), fitas e discos de memória externa. As linguagens de alto nível como o Cobol, Fortran e Algol encontraram seus grandes espaços. Em especial aqui destaco o Fortran. O Assembler (linguagem simbólica muito próxima à linguagem de máquina) era demasiadamente profissional e detalhista para ser manipulado pela grande maioria dos usuários não profissionais.

Tal tendência de simplificação das linguagens (*software*) se acentua a partir da década de 70.

Conforme o modelo da máquina analítica de Babbage, constituída de entrada, processamento e saída de dados, e considerando as definições de *hardware* e *software*, é

possível identificar o papel desempenhado pelas linguagens de programação a partir da arquitetura dos computadores (Figura 6).

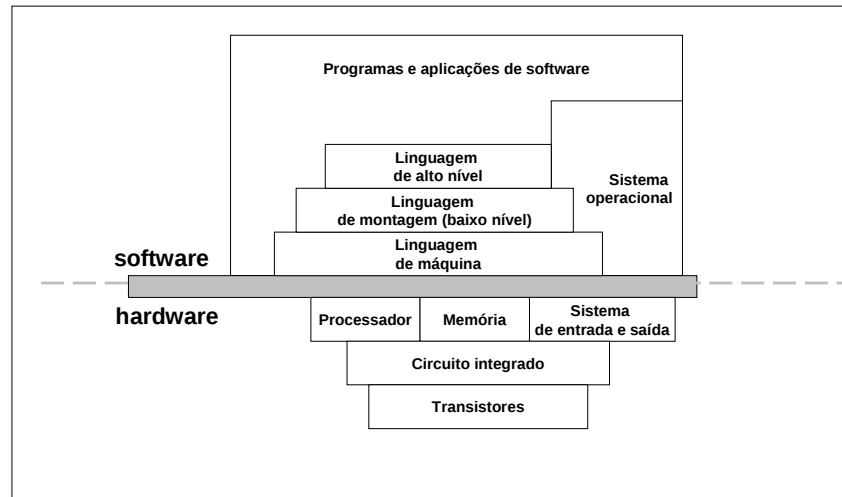


Figura 6 – Arquitetura básica de *hardware* e *software* (Desenvolvido com base em PACITTI, 1988).

O processador é o item de *hardware* que representa o centro do processamento das informações. O isolamento do centro de processamento só passou a fazer sentido a partir dos circuitos integrados e *microchips*, que se constituíram em uma parcela de *hardware* responsável por processar as informações. A memória é a parte do *hardware* responsável pelo armazenamento de informações. Pelo projeto lógico do computador definido por Von Neumann, as instruções deveriam ser armazenadas na memória do computador ao invés de serem lidas a partir de cartões perfurados para então serem executadas, isto porque a arquitetura estabelecida por Von Neumann descreve como padrão: entrada de dados, processamento com armazenamento (memória) e saída da informação processada. O sistema de entrada de informação, nas duas primeiras gerações de computadores, operava a partir de cartões perfurados e chaveamento de circuito por interruptores e fios.

A partir da evolução do *hardware* foi possível, com o uso das linguagens de programação, o desenvolvimento de *softwares* específicos para cada necessidade. Em

alguns computadores da segunda geração e em praticamente todos os computadores da terceira passa a existir o chamado **sistema operacional**, uma camada de *software* que opera entre o *hardware* e os programas aplicativos voltados ao usuário final. Trata-se de uma estrutura de *software* ampla, que incorpora aspectos de baixo nível (como *drivers* de dispositivos e gerência de memória física) e de alto nível, como programas utilitários e a própria interface gráfica (PATTERSON & HENESSY, 2005).

Linguagem é uma maneira de comunicação que segue uma forma e uma estrutura com significado interpretável. A linguagem utilizada no cotidiano é diferente da linguagem utilizada pela máquina. A máquina trabalha somente com códigos numéricos (linguagem de máquina), baseados nos números 0 e 1 (sistema binário), que representam impulsos elétricos, ausente e presente. Assim, qualquer linguagem de programação deve estar situada entre dois extremos: o da linguagem natural do homem (muito clara, porém mais lenta de ser interpretada pela máquina) e o da linguagem de máquina (muito rápida, porém complexa) (Gonick, 1986). Linguagens de alto nível são estruturadas e de mais fácil entendimento que as linguagens de baixo nível (linguagens de máquina) para a programação profissional e não profissional.

O sistema operacional prepara toda a infra-estrutura necessária, tal como o reconhecimento de todos os componentes de *hardware* que compõem o computador, para que possam ser utilizados harmonicamente pelo usuário sem que seja necessário o conhecimento específico em cada um deles para que o computador seja operado. Em outras palavras, ao ser ligado, o sistema operacional “reconhece” todos os componentes de *hardware* que compõem o computador, fornecendo uma interface básica ao usuário para que este possa utilizar todo o conjunto que compõe o *hardware* de maneira transparente,

inclusive utilizando outros *softwares* que podem ser instalados e conseqüentemente gerenciados pelo sistema operacional.

A metade da arquitetura equivalente ao *software* constitui-se nos elementos intangíveis através dos quais se dá a manipulação e interação humana com a máquina. A primeira geração de computadores era programada a partir da linguagem de máquina, transmitida ao computador como operações a serem executadas de forma binária. Entre as linguagens de máquina utilizadas pelos computadores de primeira geração e as linguagens de baixo nível utilizadas pelos de segunda geração, existe em comum a forte relação entre as operações implementadas na linguagem e as operações implementadas pelo *hardware*. Com as linguagens de baixo nível é necessário que o programador tenha completo conhecimento de programação para se comunicar com o *hardware* e descrever as ordens adequadas para a realização de um determinado programa para a solução de uma tarefa ou problema. Para programar em linguagem de máquina ou linguagem de baixo nível, o programador deve explicitamente determinar quais conjuntos de transístores, chamados de registradores para facilitar a lógica do programa, executarão determinada instrução. Tais particularidades da programação em linguagens de máquina ou de baixo nível são abstraídas nas linguagens de alto nível, que equacionam a complexidade do conhecimento do *hardware* por parte do programador, tornando assim a programação mais simples. As linguagens de alto nível aproximam-se das linguagens utilizadas pelo homem para expressar os problemas e algoritmos a serem processados pelo *hardware* e cada declaração de instrução em uma linguagem de alto nível equivale a várias declarações em uma linguagem de baixo nível. Por exemplo, em uma linguagem de alto nível poderiam ser descritas as instruções para validação de acesso de usuário conforme algoritmo da Figura 7.

```

Programa Soma;
  Variáveis
    Numero_1, Numero_2, Total = inteiro;

Íncio
  Escreva("Entre com o primeiro número inteiro");
  Leia(Numero_1);
  Escreva("Entre com o segundo número inteiro");
  Leia(Numero_2);
  Total := Numero_1 + Numero_2
  Escreva(" O resultado é da soma entre os números é: ", Total)
Fim.

```

Figura 7 – Exemplo de instruções em linguagem de alto nível e estruturada.

A Figura 8 representa o mesmo conjunto de instruções programadas pelo programador para os diferentes tipos de linguagem, tendo como objetivo o somatório de duas variáveis. Em programação para computadores um problema a ser resolvido – neste caso o somatório de duas variáveis – é denominado algoritmo.

Linguagem de máquina	Linguagem de baixo nível	Linguagem de alto nível
0010 0001 1110	LOAD R1, val1	Total = Numero_1 + Numero_2
0010 0010 1111	LOAD R2, val2	
0001 0001 0010	ADD R1, R2	
0011 0001 1111	STORE R1, val2	

Figura 8 – Instruções para o somatório de duas variáveis em diferentes níveis de linguagem.

Em linguagem de máquina, as seguintes instruções são necessárias para instruir a máquina com os seguintes comandos conforme a Figura 9.

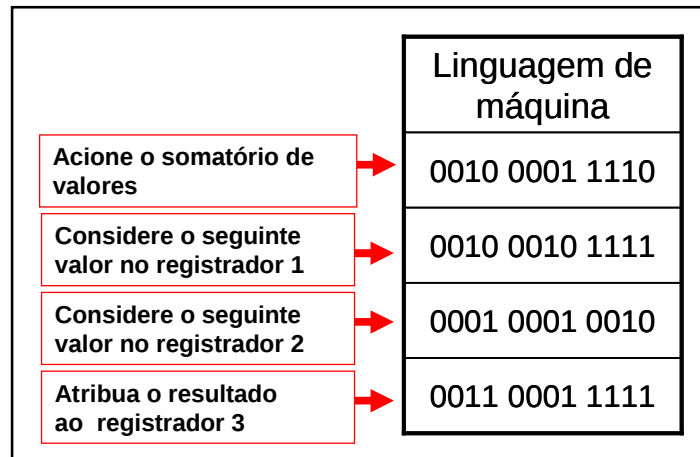


Figura 9 – Sequência de instruções para o somatório de duas variáveis em linguagem de máquina.

Com a utilização da linguagem de máquina há uma grande probabilidade de erro em todos os estágios do processo de programação. Mesmo a programação de algoritmos simples resulta em longos programas, o que dificulta o processo de validação e eventuais detecções de erros. Na linguagem de alto nível as instruções programadas pelo programador estão descritas de forma mais simples, mas é requerido do programador explicitamente o conhecimento do *hardware*, nesse caso dos registradores 1 e 2 (Figura 10).

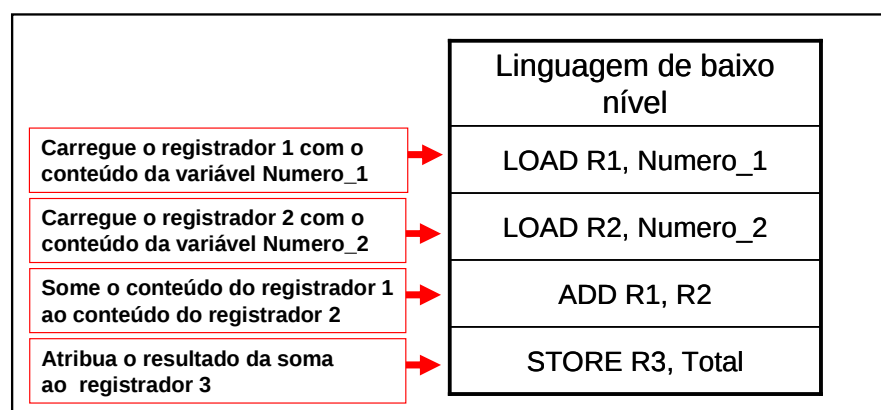


Figura 10 – Sequência de instruções para o somatório de duas variáveis em linguagem de baixo nível.

A interpretação da linguagem de alto nível pelo computador em relação à linguagem de baixo nível pode ser realizada fazendo com que o computador seja o

responsável pelo estágio de tradução. Na linguagem de máquina a representação das instruções através de códigos numéricos é extremamente cansativa e, além disso, a lógica do programa fica difícil de ser compreendida. Um programa de computador é escrito usando-se alguma linguagem de programação. Existem diversos tipos de linguagens de alto nível (ver anexo I), com muitas características distintas, mas o *hardware* do computador não entende essas linguagens diretamente, sendo necessário, portanto, um programa que as traduza para uma linguagem de mais baixo nível que seja entendida pelo *hardware*.

Na linguagem de alto nível ocorre uma total abstração em relação ao conhecimento requerido do programador em relação ao *hardware* (Figura 11), pois o programador precisa focar nas fórmulas e condições da tarefa de programação e não nos detalhes técnicos que tornam a atividade possível.

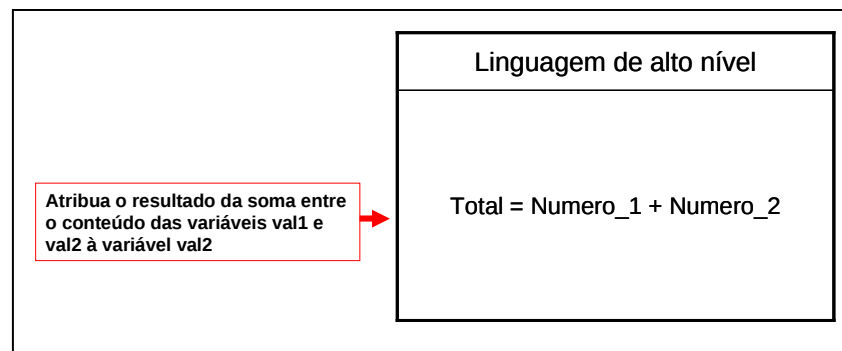


Figura 11 – Abstração do conhecimento do hardware possibilitado pela linguagem de alto nível.

As linguagens de alto nível oferecem ao programador maiores vantagens devido à baixa complexidade na descrição dos comandos em relação às linguagens de baixo nível. A principal motivação para o uso de linguagens de alto nível é que os problemas podem

ser solucionados de maneira mais rápida, devido à forma em que a linguagem permite a estruturação do programa. Para que as facilidades das linguagens de alto nível possam proceder, é necessária a existência de um programa especial, o **compilador**²⁴, para que o código escrito em linguagem de alto nível, ou seja, de maneira mais clara e estruturada, possa ser convertido para linguagem de máquina. Dessa forma, em todo o processo de produção de *software*, o compilador é o programa que efetua a conversão ou tradução do código descrito em linguagem de alto nível para o código resultante em linguagem de máquina. Figura 12 – Conversão da linguagem de alto nível para a linguagem de máquina feita pelo compilador., facilitando assim a atividade de programação para os programadores.

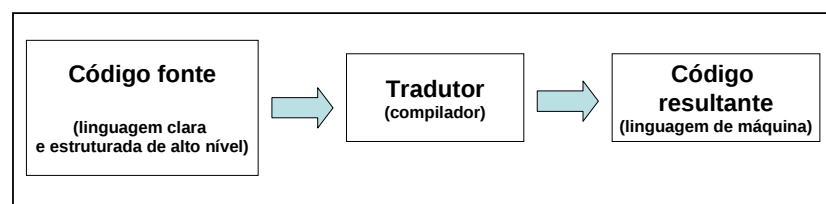


Figura 12 – Conversão da linguagem de alto nível para a linguagem de máquina feita pelo compilador.

No sentido geral, compilador é qualquer programa que transforme um conjunto de símbolos em outro, obedecendo a uma série de regras sintáticas²⁵ e traduz todo o código-fonte de programas escritos em uma linguagem de alto nível, em linguagem de máquina, preparando dessa forma o código resultante para a execução do programa. Cada linguagem de programação possui regras sintáticas características e finalidades distintas. Dessa forma, desde a primeira geração de computadores, foram criadas diferentes linguagens e conseqüentemente diversos compiladores. A partir da segunda geração

²⁴ “O programa tradutor é geralmente chamado de compilador, e é responsável pela transformação de um programa escrito numa linguagem de alto nível em um programa expresso numa linguagem de baixo nível” (SCHMIDT, 1986, p. 46)

²⁵ Análise sintática (também conhecida pelo termo em inglês *parsing*) é o processo de analisar uma sequência de entrada (lida de um arquivo de computador ou do teclado, por exemplo) para determinar sua estrutura gramatical segundo uma determinada gramática formal (TANENBAUM, 2001).

(*Fortran, Cobol, Algol*), tais linguagens adotaram uma sintaxe de nível mais alto, que pode ser mais facilmente entendida por programadores humanos.

CAPÍTULO 3

PRODUÇÃO DE *SOFTWARES* E A SUBSUNÇÃO DO TRABALHO INTELECTUAL

As áreas de manufatura e ciência da computação influenciam fortemente os avanços tecnológicos relacionados à evolução de *software* para sistemas de manufatura e automação industrial. As unidades de controle computadorizadas são integradas aos sistemas de manufatura a partir de interfaces de *software* que facilitam essa integração, as quais apresentam requisitos de conexão física (*hardware*) e lógica (*software*) entre os componentes. Nesse conjunto são requeridos *softwares* adicionais de controle de integração. Os *softwares* de controle especificam os procedimentos de entrada e saída das unidades de controle das máquinas, que dirigem as operações dos dispositivos de manufatura. Todas as combinações das entradas da unidade de controle, que são significativas ao processo controlado, devem ser capturadas a partir da interface de *software*. A produção e desenvolvimento dessas interfaces, que se caracterizam como produtos de *software*, são consideradas de nível de abstração complexo e de difícil interpretação. Para a sua produção, várias linguagens de programação, com níveis de abstração altos são utilizadas. O desenvolvimento de *software* de controle para sistemas de manufatura é considerado como uma tarefa difícil, principalmente devido à complexidade que esses sistemas apresentam tanto em nível conceitual como de implementação. Além disso, o desenvolvimento do *software* de controle representa um custo alto no que diz respeito ao desenvolvimento de sistemas de controle de chão-de-fábrica.

A utilização de esquemas de representação para a produção de *software* convenientes e ambientes de desenvolvimento que possibilitem a criação de sistemas de controle flexíveis de manufatura de fácil utilização, é importante na busca da diminuição

dos custos de desenvolvimento e da complexidade de implementação desses sistemas. Yourdon (1984) define a metodologia de desenvolvimento de *software* como uma especificação formal de um sistema para construir sistemas. Tal metodologia define as partes componentes de um sistema para a construção de sistemas de informação computadorizados, ou seja, as fases ou atividades que existem em um projeto de desenvolvimento de *software* típico. No que diz respeito ao desenvolvimento de *software* de controle para sistemas de chão-de-fábrica, esquemas de representação são usados com o objetivo de modelar tanto o sistema físico quanto a interação entre sistemas de aplicação.

Os sistemas de automação flexível evoluíram a ponto de permitir a completa interação dos diferentes ofícios de trabalho, envolvidos ao longo do processo produtivo da indústria. Com a utilização de máquinas de automação flexível, tais como CNC, neste tipo de indústria, as fases de concepção (ou fase de projeto, *design*) e execução (ou fase de manufatura), se tornam ainda mais distintas, devido ao uso da modelagem assistida por computador. Com o auxílio desse tipo de ferramenta, é possível retirar do chão de fábrica não somente a fase de concepção, mas também parte da execução propriamente dita, partindo-se do princípio de que o *software* concebido na fase de projeto (*design*) é embutido nas máquinas do chão de fábrica de forma a comandar a execução.

A Tabela 1 mostra a evolução dos sistemas CAD conforme o desenvolvimento dos computadores:

Tabela 1 – Evolução dos sistemas CAD conforme a evolução dos computadores e linguagens de programação (FELDENS, 2000, p. 8) e anexo I, evolução das linguagens de computação (LÉVÉNEZ, 2007)

<i>Evolução das ferramentas CAD</i>	<i>Evolução dos computadores</i>	<i>Evolução das principais linguagens de programação</i>
CAD em duas dimensões (2D) • 1957 – CAD PRONTO – Hanrathy – GE • 1960 – Sketchpad – Sutherland – MIT	• 1947 – ENIAC – 10 Kflop US\$ 4 milhões • 1955 – IBM 1620 – 200 Kflop • 1964 – IBM 360 – 1 Mflop • 1975 – PC – 10 Mflop • 1990 – Pentium – 100 Mflop • 2005 – Pentium 4 – 300 Mflop – US\$ 1.000	• FORTRAN novembro de 1954, FORTRAN I outubro de 1956 e ALGOL58 em 1958. • COBOL em 1965
CAD em três dimensões (3D) • 1981 – Unisolid • 1982 – Catia e I-deas • 1983 – Intergraph • 1987 – Pro-engineer (PTC) • 1995 – Solidworks (Dassault) • 1996 – SolidEdge (Intergraph) e Mechanical Desktop (Autodesk)		• Smalltalk-80 em 1980 • Linguagem C com Classes em abril de 1980 • Linguagem C++ em julho de 1983

Os sistemas CAD, que possuem o conceito de auxílio computadorizado à concepção, abrangem toda atividade em que está presente o uso do computador para desenvolver, analisar ou modificar o projeto de um produto ou processo. Os sistemas CAD mais atuais baseiam-se em computação gráfica interativa (ICG, *Interactive Computer Graphics*), através da qual o sistema se comunica com o usuário. Nesses sistemas o computador é empregado para criar, transformar e mostrar dados na forma de figuras e símbolos. O operador do sistema, que no caso pode ser o próprio projetista responsável pela concepção do produto, comunica dados e comandos para o computador através de dispositivos de entrada adequados (*interface*), a partir dos quais o computador cria uma imagem na tela do vídeo, utilizando rotinas de comandos gráficos. VIEIRA (1995), observou o processo de inovação tecnológica de uma indústria têxtil instalada no Brasil,

com o objetivo de identificar o elenco de variáveis tecnológicas, econômicas, sociais e políticas envolvidas no processo decisório em um sistema de modelagem e planejamento de corte assistido por computador. A partir do estudo, foi possível verificar a organização típica de indústrias que fazem uso de ferramentas CAD/CAM e máquinas CNC, atentando para a importância dos artefatos computacionais e ferramentas assistidas por computador. Conforme apresentado no estudo, as fases do processo de produção estão divididas em planejamento da produção ou planejamento da talharia, engenharia de produto e produção. A engenharia do produto é a fase em que se organiza o insumo da produção (malha) e se determina o rigor das especificações técnicas de forma a atender às grades de composição dos pedidos da clientela. O planejamento do produto é a fase em que se dá a transformação fundamental do produto semi-elaborado, tanto na sua produção, quanto em seu controle, promovendo a integração entre as fases de engenharia do produto e produção, a partir da utilização de tecnologias de computação gráfica e planejamento assistido por computador, com a utilização de Bancos de Modelos – base de dados com um conjunto de modelos gerados pela equipe de modelagem – para estudar a melhor forma de cortar os tecidos, segundo as ordens de produção de talharia. Essa fase é subdividida nas subfases de planejamento da coleção e projeto da coleção, nas quais são realizadas as atividades de identificação no mercado das oportunidades de lançamento de produtos e prototipagem²⁶ de produtos com algumas de suas características básicas respectivamente. Na fase de produção são executadas as atividades necessárias para a confecção do produto. O computador, assistindo a programação do corte concebido durante a fase de planejamento, a partir de dados gerados por uma instalação CAD, garante a confiabilidade da informação e precisão requerida na confecção do produto. As

²⁶ Prototipagem é uma técnica baseada numa visão evolutiva do desenvolvimento do produto, que envolve a produção de versões iniciais – tais como maquetes para a arquitetura. Com isso, podem-se realizar verificações e experimentações para se avaliar algumas de suas qualidades, antes que o produto venha realmente a ser completamente construído (BOEHM, 1984, p. 473-484).

fases de planejamento da produção, engenharia de produto e produção descritos no estudo, podem ser melhor detalhadas conforme o ciclo genérico de projeto de produto, sintetizado por Feldens, apresentado na Figura 13:

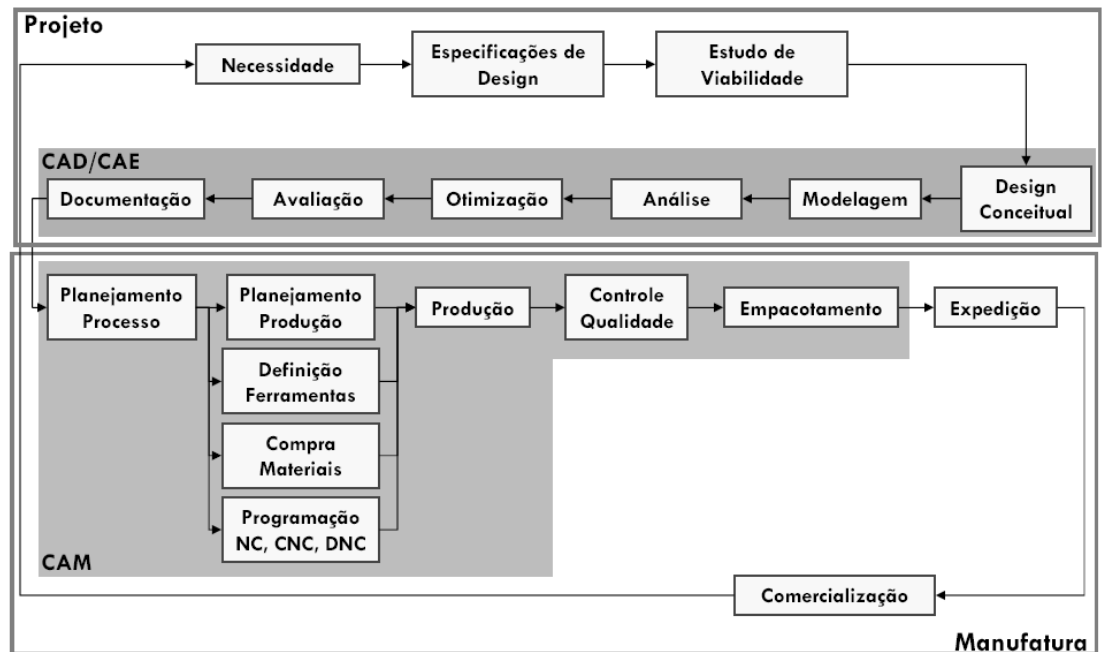


Figura 13 – Ciclo de projeto de produto (FELDENS, 2000, p. 6).

A fase de projeto tem início com o estudo de viabilidade do produto ou conforme contratos estabelecidos entre a empresa e demandantes do mercado. A origem da demanda está relacionada à natureza do produto e seu respectivo uso. Para cada novo produto, uma vez definida a demanda, é iniciada a fase de concepção.

Em todo o sistema CAD/CAM existe um elemento chamado célula, que constitui o posto onde o trabalhador executa o seu trabalho, também denominado posto ou estação de trabalho (*workstation*). Existem sistemas que suportam inúmeras estações de trabalho interligadas a uma base de dados única e sistemas constituídos por uma única estação. A ferramenta CAD/CAM está a serviço do trabalhador (*designer*), que não necessita possuir

profundos conhecimentos de informática. Para isso, conta com um conjunto de *hardware* que, em conjunto com uma comunicação de alto nível proporcionada pelo *software*, fornece todos os subsídios necessários para interagir com a unidade *hardware* e *software* da maneira mais simples possível. Tal simplicidade é possível graças ao aplicativo (*software*) e periféricos (*hardware*) integrados através da unidade central de processamento – CPU. Conforme Castelltort (1988, p. 12), os comandos de fácil compreensão não têm

nada a ver com a função de programar o computador através de uma determinada linguagem. O engenheiro ou profissional de *design* usa a ferramenta CAD/CAM para desenvolver seu trabalho. Da mesma forma que não é preciso saber como se fabrica um carro para ser um bom motorista, tampouco é necessário saber programar computadores para poder mexer num sistema CAD/CAM e beneficiar-se de seus altos serviços.

No sistema CAD/CAM, o *hardware* constitui a parte física do sistema e é constituído pela CPU, terminal de visualização de alta resolução, mesas digitalizadoras, dispositivos magnéticos de armazenamento para a constituição de banco de dados de alta capacidade, impressoras etc. O sistema CAD/CAM permite flexibilidade de *hardware*, ou seja, a partir das necessidades do trabalhador ou da natureza do produto projetado, podem ser adotados diferentes tipos de periféricos. Esse princípio sugere a necessidade de existência de padrões que permitam a comunicação da CPU com os diferentes tipos de periféricos. O estabelecimento de padrões de *hardware* integrados por diferentes tipos de *software* foi crucial para o desenvolvimento relativamente acelerado (aproximadamente quatro décadas, Figura 11) dos sistemas CAD/CAM.

A parte lógica do sistema CAD/CAM é constituída pelo *software*. O *software* que integra o sistema lógico comanda o sistema através da transferência de dados e o processamento de informações. O conjunto de *software* é responsável pela tarefa de

geometria, adição, modificação, cálculos matemáticos etc. Sob o ponto de vista pragmático, o *software* que integra o sistema CAD/CAM atribui sentido ao conjunto de periféricos de *hardware*, comunicando-os entre si e com a unidade central, com o intuito de proporcionar ao trabalhador a elaboração do produto físico, percebê-lo e testá-lo, virtualmente, antes que este esteja constituído fisicamente. A partir desse ponto, todo o processo de concepção é desenvolvido com o auxílio de ferramentas CAD/CAE que incorporam recursos de realidade virtual, inteligência artificial e cálculos matemáticos, que automatizam todo tipo de tarefa repetitiva ligada ao trabalho intelectual envolvido com a fase de projeto correspondente à concepção criativa. É nesse ponto que a difusão da base técnica e microeletrônica, enfim, abre novas possibilidades de articulação dos agentes na produção.

Considerando o início do trabalho de produção industrial na fase de projeto (concepção ou *design*) do produto, o *software*, que constitui parte do sistema CAD/CAM, assume o papel de ferramenta de modelagem e direcionador da concepção criativa do trabalhador. Essa fase é desenvolvida e integrada virtualmente através das ferramentas CAD/CAM. Castelltort apresenta um exemplo de modelagem de elementos finitos (*finite element modelling*), técnica que pretende simular o comportamento de um objeto (com geometria previamente definida no CAD) com determinada estrutura e material. Conforme Castelltort (1988, p. 42) pode-se submetê-lo a testes térmicos, esforços mecânicos, de resistência etc., a fim de prever seu comportamento na vida real. Nota-se a extrema virtualização da fase conceitual de um processo industrial que faz uso do sistema CAD/CAM

Imaginemos um projeto de uma refinaria de petróleo. Nele intervém múltiplas disciplinas, tais como arquitetura, engenharia, mecânica, hidráulica, química etc., com seus respectivos projetos de estruturas, equipamentos especiais e maquinaria, redes elétricas e tubulação etc. Cada um deles será desenvolvido em uma

Workstation pela equipe correspondente. Desta forma, em cada estação gráfica far-se-á uma parte do projeto.

Quando as workstations estão interligadas, pode-se gerenciar todos os subprojetos de uma só vez, supervisionando seu estado em qualquer momento e verificando possíveis interferências entre eles. A soma das partes constituirá a maquete eletrônica que, além das informações próprias do projeto, terá todas aquelas associadas aos componentes de cada subprojeto. Toda essa informação possibilitará conhecer de antemão todos os itens na hora do empreendimento.

Uma vez que todos os subprojetos estão “montados” e a maquete está pronta [virtualmente], podemos analisar todos os detalhes, obtendo qualquer vista de qualquer localização. Este tipo de visualização de grande realismo nos permite até caminhar pelo interior da maquete e nos situar em locais de difícil ou impossível acesso na realidade.

Conforme apontado pelo autor, durante a fase de projeto (*design*) do produto, o *software* assume o papel de ferramenta direcionadora da concepção criativa do trabalhador, além de integradora das diversas disciplinas e controle do trabalho. Nesse ponto há uma tendência de codificação da criação, por intermédio do *software*, que além desta característica maximiza o nível de controle da gerência que se utiliza de ferramentas e técnicas computacionais de forma coordenada, em um processo controlado, a fim de se produzir um produto (virtual) padronizado.

Durante a fase de concepção, as diversas disciplinas se agrupam em estações de trabalho, as quais compartilham uma base de dados única a fim de desenvolverem um produto em conjunto. A base de dados das ferramentas CAD/CAM possui bibliotecas de componentes e peças utilizadas por diversos fabricantes em diferentes tipos e padrões, todas dispostas ao trabalhador virtualmente de maneira a ser possível elaborar protótipos de produtos virtuais em diferentes tipos, preços e padrões. Nesse sentido, a base de dados é a fonte de informações com a qual o trabalhador interage durante todo o processo de trabalho, Figura 14.

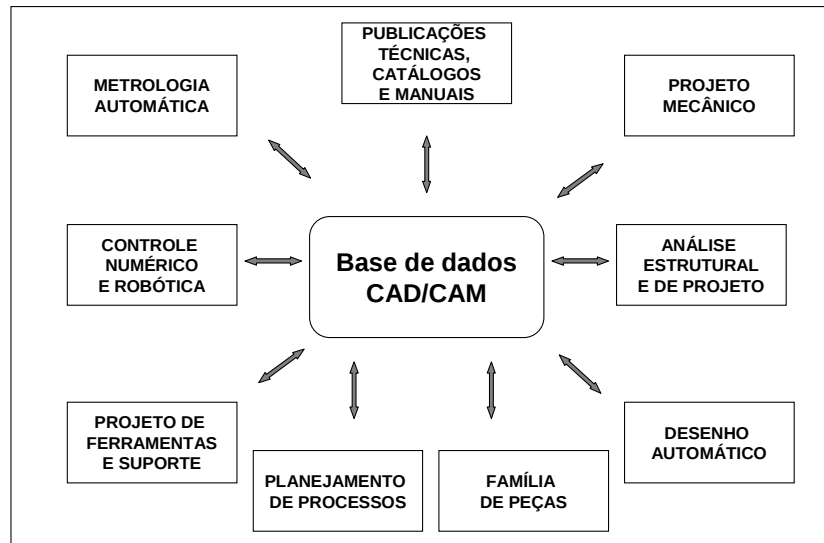


Figura 14 - CAD/CAM: Sistema Integrado de Produção. UFMG (DEMEC/EE/UFMG - <http://www.demec.ufmg.br/Grupos/Usinagem/CADCAM.htm>)

Nesse sentido, o trabalho produzido por cada disciplina de trabalho, ou agrupamento dos ofícios dos trabalhadores, em cada estação de trabalho pode ser controlado. A partir da Figura 15 é possível verificar de que forma ocorre a interação entre as estações de trabalho e o compartilhamento das informações oriundas da base de dados centralizada. Nesse contexto é possível ter uma noção de como o processo de trabalho, envolvendo tais elementos, pode ser controlado.

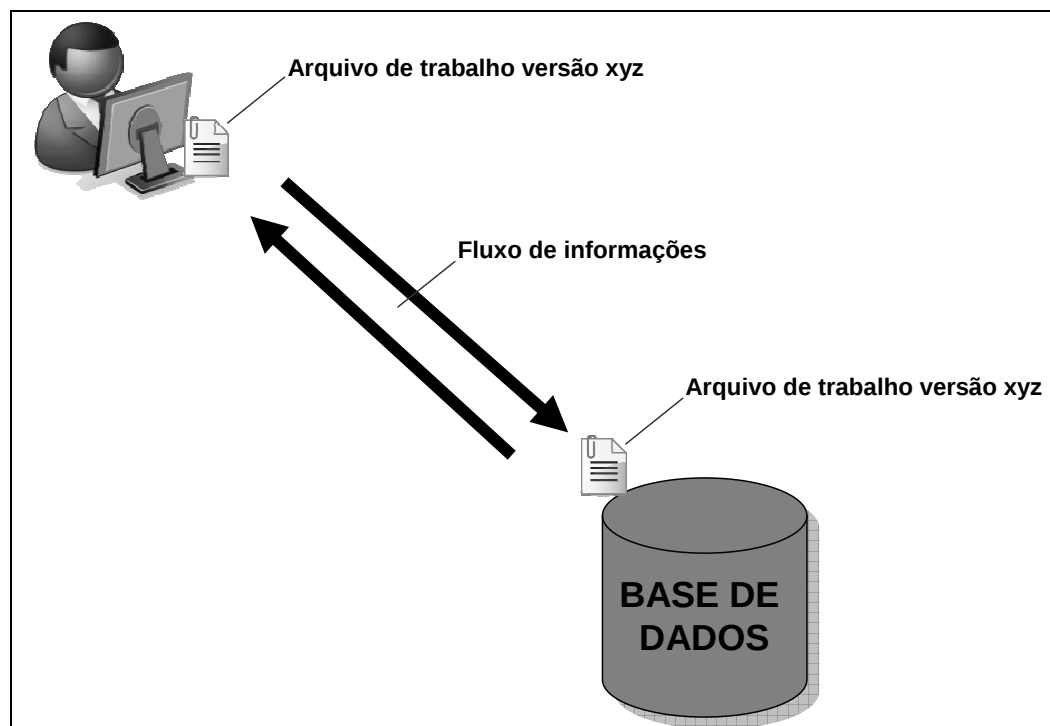


Figura 15 – Processo de produção com ferramentas cad/cam (CASTELLTORT, 1988, p. 42 e 43).

Para trabalhar em seu posto através do sistema CAD/CAM, o trabalhador interage com a ferramenta efetuando as devidas modificações no arquivo. Por sua vez, ao “manipular” o arquivo, o trabalhador acessa as informações armazenadas na base de dados. Tais informações servem de insumo para que o trabalhador possa desenvolver as tarefas relativas às disciplinas. Considerando que o produto gerado pelas equipes multidisciplinares é um arquivo contendo todas as informações do escopo do trabalho relativo à disciplina, o sistema CAD/CAM controla tal interação e cooperação entre as equipes de forma automática. Cada arquivo utilizado pelos trabalhadores integrantes das equipes pode ter a sua versão, que basicamente consiste no histórico de desenvolvimento do documento, controlada a ponto de identificar qual trabalhador²⁷ efetuou a última modificação no arquivo. O resultado ou produto das tarefas de uma disciplina se traduz

²⁷ No contexto informático convencionase que toda pessoa que interage com o *software* seja denominada de usuário.

em um arquivo ou um conjunto de arquivos que, aliado aos outros arquivos das demais disciplinas envolvidas resultam no produto final, aquele esperado no início do processo.

Durante todo o processo de trabalho, o controle das atividades pode ser monitorado pela gerência a partir da verificação do *status* de desenvolvimento dos diferentes arquivos das disciplinas envolvidas no projeto. Através das consultas de verificação, efetuadas pela gerência, é possível checar o status de desenvolvimento das tarefas de cada trabalhador, além de ser possível à verificação prévia dos erros técnicos nos arquivos gerados (BALDAM & COSTA, 2009). Nesse sentido, os controles disponibilizados à gerência a partir da ferramenta, além do acompanhamento das atividades de um grande número de trabalhadores de forma simultânea, permitem que todo o trabalho de controle seja feito independentemente de local, bastando apenas que a base de dados esteja acessível à gerência pelas redes telemáticas, assim como seus respectivos relatórios, potencializando dessa forma a capacidade de controle do trabalho durante todo o processo. Conforme Braverman (1974, p. 163), mais ainda que a velocidade proporcionada na produção, “o elemento fundamental na evolução da maquinaria não é a dimensão, complexidade ou velocidade de operação, mas a maneira pela qual suas operações são controladas”.

Nesse sentido, o conjunto formado pela estação de trabalho dotada da ferramenta CAD/CAM equivale à máquina-ferramenta, estabelecendo ritmos e condições que garantem o maior controle do trabalho. Em Braverman (1974), as máquinas-ferramenta se desenvolvem a tal ponto que extrapolam o complemento de movimentos precisos e suplemento dos músculos do trabalhador e se concretizam em autômatos capazes de executar tarefas ininterruptas em ciclos definidos de produção, os quais “seguem uma

seqüência de operações que podem, basicamente, fazer apenas aquilo que foi projetado e para que foram construídas, ou adaptadas a uma variedade limitada de funções pela alteração interna (eixos e engrenagens) dos dispositivos²⁸”.

Podemos entender o processo, no seu conjunto, dividido em três momentos. O segundo momento é aquele em que a célula composta pela estação de trabalho (*hardware*) e a ferramenta CAD/CAM (*software*) potencializam o processo produtivo virtualizando a fase de concepção do produto, intelectualizando o trabalho através do *software*, gerando como resultado um processo de produção extremamente automatizado. O terceiro momento, após a concepção do produto, é o das adaptações citadas pelo autor na maquinaria automatizada, durante a fase de produção no chão de fábrica, que são realizadas por meio de ajustes através do *software* de controle das máquinas (*hardware*) e não mais pela “alteração interna (eixos e engrenagens) dos dispositivos”. Dessa forma, todo o trabalho envolvido para que a máquina-ferramenta automatizada (uma MFCNC, por exemplo) possa produzir aquilo que fora projetado durante a fase de concepção nas estações de trabalho, é intelectualizado. O *software* assume papel distinto e fundamental tanto na fase de concepção do produto, através do CAD/CAM, quanto na fase de produção controlando as máquinas automatizadas.

Mas há o momento que antecede a fase de projeto do produto industrial específico (primeiro momento, portanto) e se configura como uma fase distinta de produção, a **produção de *software***, como por exemplo, a fase de produção do *software* CAD/CAM. Nesse tipo de produção, um *software* assume o papel de ferramenta de concepção e desenvolvimento de um produto que é também um *software*. Em outros termos, os

²⁸ Nessa linha de raciocínio o autor cita como exemplo a máquina de lavar e o seu conjunto finito de programações possíveis (BRAVERMAN, 1974).

sistemas CAM são construídos a partir de ferramentas de *software* cujo processo de desenvolvimento é marcado por fases similares às da produção industrial automatizada: concepção e execução.

Assim, as ferramentas (*softwares*) CAD/CAM são desenvolvidas a partir de ferramentas similares, no sentido taxionômico, chamadas CASE, voltadas para o desenvolvimento de *software*. Estas têm um potencial ainda mais amplo, pois permitem a concepção de qualquer tipo de *software*, inclusive os utilizados nas máquinas CNC que possibilitam a programação e reprogramação para a produção de peças de maneira flexível.

Frederick Brooks (1986), assume a inexistência de qualquer ferramenta ou técnica que possibilite um avanço na concepção e desenvolvimento de *software*, ou na simplicidade e confiabilidade desse no sentido de se obter um processo automatizado de produção, similar a um processo de produção industrial. O autor considera que o surgimento de algum princípio que permita um avanço no sentido de solucionar os maiores problemas no desenvolvimento de *software* não é necessariamente impossível, mas improvável, conforme as dificuldades existentes durante o processo de desenvolvimento de *software*, as quais classifica como essenciais e acidentais. A compreensão dos fatores que permitem a análise da complexidade que envolve o processo de produção de *software* pressupõe a compreensão da sua natureza e de como vem sendo tratado ao longo dos anos. Brooks aponta a complexidade, conformidade, sujeição a mudanças e invisibilidade, como aspectos básicos que caracterizam a natureza de qualquer *software* e classifica ainda os problemas enfrentados pelo trabalhador de *software* como arbitrários.

Os problemas de concepção e desenvolvimento voltam-se para o trabalhador de *software* a partir de instituições, pessoas e sistemas humanos aos quais o *software* deve adequar-se, sendo tal conformidade dinâmica, visto que os sistemas humanos, como as leis, enfrentam mudanças eventuais, cabendo ao *software* adequar-se à realidade vigente. Além disso, o crescimento de um *software* não acompanha o crescimento de qualquer outra construção humana em conformidade com princípios fundamentais e estáveis. Aumentar um *software* implica em aumentar a quantidade e variedade das características das entidades desse e as relações entre essas entidades, enquanto aumentar qualquer outra construção humana física, susceptível a princípios fundamentais estáveis, implica em aumentar o tamanho de suas características, ou apenas a quantidade dessas. Nesse sentido, a complexidade de um *software* aumenta de forma não linear com o seu crescimento. A intangibilidade do *software* só permite a sua aproximação do mundo real, sujeito às leis da física, quando interpretado a partir de metáforas.

A complexidade de desenvolvimento de *software* é similar à complexidade que afeta outras áreas de conhecimento, como por exemplo a física, que lida com objetos e corpos grandes e complexos que precisam eventualmente ser examinados ao nível de partículas. Entretanto, a física adota princípios e leis facilitando a abordagem na investigação e solução de problemas complexos, ou seja, busca, a partir de leis básicas e princípios unificadores, a racionalização de fenômenos conhecidos, como base para a previsão e construção de novos fenômenos. Nos corpos e objetos sujeitos às leis da física que compõem o mundo material, os princípios unificadores tendem a ser estáveis. Tais princípios são fundamentais, por exemplo, para projetos de engenharia que assumem

como pressupostas leis que permitem o cálculo estrutural para a concepção e construção de complexas estruturas, seja na construção civil, naval ou eletromecânica.

Ao contrário dos projetos de engenharia, os produtos de *software* não são projetados, concebidos e desenvolvidos com base em princípios fundamentais e estáveis. Ao contrário, a complexidade que envolve o processo de produção de *software* é baseada na existência de princípios instáveis ou arbitrários (BROOKS, 1987). A complexidade do *software* é uma propriedade essencial e não acidental de tal forma que a descrição da complexidade de um artefato de *software* torna-se uma tarefa extremamente difícil, pois geralmente contém a sua própria essência. A essência de uma entidade de *software* é a concepção de conceitos relacionados, tais como conjunto de dados, relacionamento entre os algoritmos e chamadas de funções. Ao conceber o projeto de uma construção civil, um engenheiro analisa a complexidade da obra, dispondo para isso de princípios fundamentais e modelos simplificados que lhe permitem abstrair tal complexidade a ponto de que possa se concentrar na essência do objeto a ser construído. Os mesmos princípios e modelos de cálculo poderão ser utilizados tanto na construção de uma ponte quanto na construção de um túnel. Por três séculos, a matemática e as ciências físicas fizeram grandes avanços na construção de modelos simplificados de fenômenos complexos, derivando as propriedades dos modelos e verificando tais propriedades através de experimentos. Tal abordagem não funciona quando as complexidades são a própria essência do fenômeno, fato que ocorre na concepção e conseqüentemente no desenvolvimento de *softwares* (BROOKS, 1987).

Outra dificuldade no desenvolvimento de *software*, classificada como acidental, é a que surge mas não necessariamente acontece. As entidades de um *software* são mais complexas do que qualquer outra construção humana, porque cada uma é única. Muitas das dificuldades acidentais foram vencidas através dos avanços tecnológicos, tal como linguagens

de programação de alto nível, compartilhamento do tempo de desenvolvimento de *software* através do escalonamento de atividades e ferramentas em redes telemáticas e ambientes de desenvolvimento unificados. Tais tecnologias reduzem a incidência de erros durante a fase de desenvolvimento de *software* e otimizam o tempo do trabalhador desenvolvedor. Entretanto, é na fase de concepção (*design*) que reside a complexidade de construção do *software*, pois durante essa fase se faz necessária **atividade criativa**.

Similarmente à abordagem de Brooks, Fernandes (2007) atribui a dificuldade na engenharia de *software* a questões ainda não plenamente definidas quanto à própria natureza do *software*. Para o autor, considerando o processo de produção de *software* como um processo misto entre manufatura e serviço, embora a estratégia de operações de serviços siga uma abordagem similar à utilizada para operações de manufatura, a própria natureza dos serviços requer algumas modificações, tais como a natureza das operações, os objetivos no desempenho das operações e decisões estratégicas relativas à infraestrutura da operação. A partir da sua abordagem, é possível perceber que o processo de produção de *software* assume caráter complexo na medida em que o produto desenvolvido assume características de produção que podem ser otimizadas com base em processos semelhantes aos da indústria, como produção fragmentada e coordenada, ao mesmo tempo em que assume características de serviço, em que o contato direto com o usuário final (demandante do *software*, no caso) se faz necessário durante o processo de produção.

Um exemplo da complexidade do processo de produção de *software* é o desenvolvimento de sistemas de controle de chão-de-fábrica, que se apresenta, no início do século XXI, como um dos maiores problemas na criação de indústrias de manufatura flexível, automatizadas ou semi-automatizadas. Uma boa parcela dessa dificuldade pode

ser atribuída aos altos custos envolvidos no desenvolvimento e manutenção do *software* de controle e à dificuldade de se conseguir definir a forma de integração dos sistemas de chão-de-fábrica (SMITH, 1992). Os dispositivos de produção, como robôs, máquinas-ferramenta e outros equipamentos de produção, assim como os computadores e redes de comunicação, geralmente são encontrados com facilidade. Entretanto, o *software* de controle necessário para a implementação de um sistema de controle flexível e integrado desses equipamentos não está prontamente disponível. Esses sistemas normalmente são multidisciplinares envolvendo conhecimento não só em manufatura, mas também em programação de computadores, análise e especificação de sistemas e redes de computadores. Por exemplo, estima-se que o custo de desenvolvimento do *software* no caso de sistemas flexíveis de manufatura gira em torno de 40% a 60% do custo total de todo o sistema automatizado (AYRES, 1989).

Os sistemas de *software* geralmente possuem uma quantidade elevada de elementos distintos, ou seja, são criados para atender a uma grande quantidade de requisitos e necessidades que precisam ser mapeados pelos trabalhadores de *software* durante a fase de concepção, através da sua interação com todos os envolvidos, a fim de serem traduzidos em *software* através das linguagens de programação pelos trabalhadores desenvolvedores. Os participantes no processo de construção do *software* são pessoas e organizações ativamente envolvidas no processo ou cujos interesses podem ser afetados como resultado da execução ou do término do projeto. Os envolvidos podem exercer influência sobre os objetivos e resultados do projeto de *software* (PMBOK GUIDE, 2004).

Para melhor ser analisado e concebido, o *software* é dividido em partes. As partes semelhantes costumam ser agrupadas em métodos, módulos, funções e classes. O conceito

de **classe** está ligado ao atual paradigma de programação intitulado **programação orientada a objetos**. A definição de classes e seus inter-relacionamentos é o principal resultado da etapa de projeto de *software*. Em geral, esse resultado é expresso em termos de alguma linguagem de modelagem de *software*. A orientação a objeto é um conceito que está relacionado com a idéia de classificar, organizar e abstrair coisas do mundo real. O termo “orientação a objetos” significa organizar o mundo real como uma coleção de objetos que incorporam estrutura de dados e um conjunto de operações que manipulam esses dados. Uma classe é um gabarito para a definição de objetos. Através da definição de uma classe descrevem-se quais propriedades (ou atributos) um objeto terá. Além da especificação de atributos, a definição da classe também descreve qual o comportamento dos objetos dessa classe, ou seja, quais as funcionalidades que podem ser aplicadas aos objetos dela. Essas funcionalidades são descritas através dos métodos. Uma classe é uma coleção de métodos da mesma categoria que compartilha as variáveis da classe (atributos). É através da modelagem de objetos que (praticamente) toda a concepção do *software* ocorre, a partir de aplicações desenvolvidas com linguagens de programação específicas para a programação orientada a objetos (SANTOS, 2003).

Na medida em que o *software* possui partes (módulos ou objetos) distintas e complexas, torna-se mais difícil a sua concepção devido a cada parte possuir caráter único, ao contrário de outras construções humanas, como prédios e automóveis (BRYANT, 2000). Nesse sentido, várias definições foram atribuídas à engenharia de *software*, como um conjunto de métodos e técnicas para a correta construção de um produto de *software*. Embora várias atribuições tenham sido feitas, a engenharia de *software* freqüentemente irá abranger o conjunto de três elementos fundamentais: métodos, ferramentas e procedimentos. Os métodos descrevem a ordem de construção do

software, as ferramentas fornecem o suporte automatizado ao método e os procedimentos definem a relação entre métodos e ferramentas com o intuito de estabelecer um processo de desenvolvimento eficiente (PRESSMAN, 1995).

Na segunda metade do século XX, o *hardware* (computador) teve o seu custo gradualmente reduzido, enquanto que o custo relacionado com o *software* aumentou, tornando-se o principal item no orçamento de sistemas²⁹, devido principalmente à complexidade dos problemas a ser resolvidos pelo *software*. Conforme Brooks (1987, p. 6), “o software, por sua vez, é apenas pensamento, o que o torna infinitamente maleável. Isso é notado pelas pessoas de um modo geral, que naturalmente pressionam por mais mudanças, por considerarem que as mesmas terão um custo reduzido”.

A qualidade do *software* produzido a partir da segunda metade do século XX não acompanhou o ritmo de evolução do *hardware*, devido basicamente a problemas no processo de desenvolvimento, acarretando como consequência estimativas de prazos e custos imprecisos. Nesse período de evolução do *hardware* é intensificada a necessidade de uma abordagem de engenharia ao desenvolvimento de *software*, no intuito de melhorar gradativamente a produtividade com o acúmulo de conhecimento que possa garantir que tarefas, dados, pessoas e tecnologia estejam alinhadas para produzir sistemas efetivos e eficientes (MURPHY, HANKEN E WATERS, 1999).

O processo de desenvolvimento de *software* abrange um conjunto de etapas, que constituem os paradigmas da engenharia de *software*, denominado **modelo de ciclo de vida de software** (PRESSMAN, 1995). Dos paradigmas mais utilizados, como o modelo

²⁹ Neste sentido é considerado sistema o conjunto de *hardware* e *software* que compõe o produto de computação orçado pelas empresas.

incremental, o concorrente e o prototipado, destaca-se o modelo do ciclo de vida clássico ou cascata (*waterfall*). Os paradigmas podem ser combinados conforme a escolha da estratégia, considerando-se a natureza do *software* a ser desenvolvido, área de aplicação, métodos e ferramentas utilizadas. A Figura 16 retrata o ciclo de vida clássico de produção de *software*.

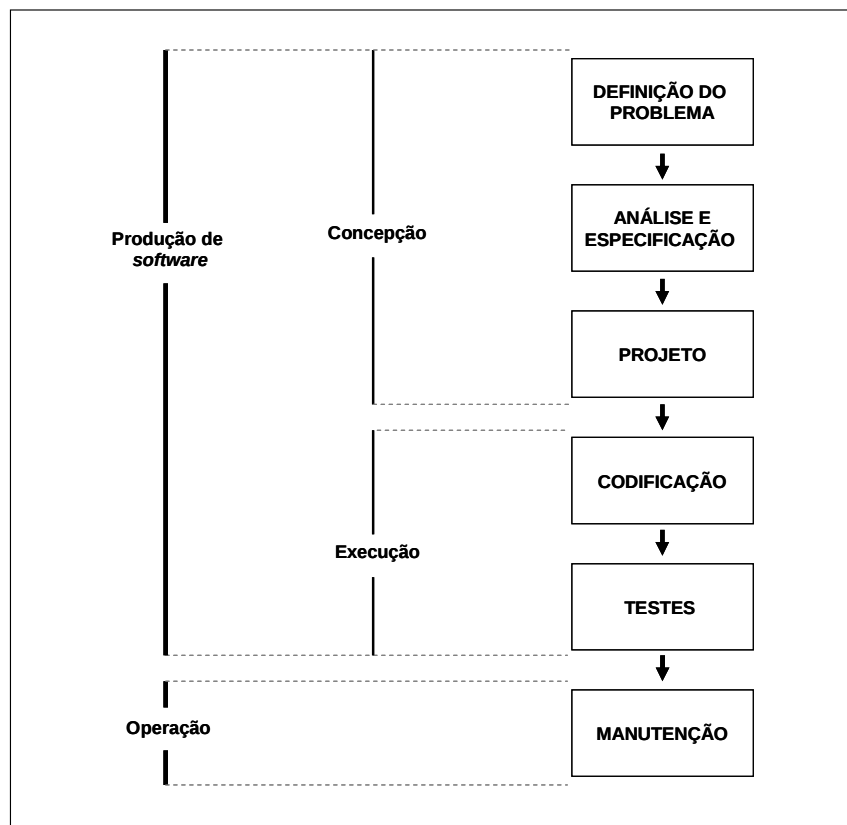


Figura 16 – Ciclo de vida clássico de software (PRESSMAN, 1995).

A fase de concepção do *software* é auxiliada pelas ferramentas CASE (*Computer-Aided Software Engineering*), as quais agregam métodos de estruturação fornecendo procedimentos e notações diagramáticas que especificam a função do *software* em diferentes níveis de abstração e permitem a construção do mesmo. A sofisticação dos métodos leva conseqüentemente a uma complexidade maior no gerenciamento do processo de desenvolvimento de *software*. As ferramentas entram no processo como

agentes que pretendem simplificar as ações envolvidas. No mercado existem diversas ferramentas CASE que agregam modelos existentes e combinações ou criam novos modelos aproveitando os melhores conceitos das metodologias existentes. Dentre as diversas metodologias existentes no mercado, tais como *Cleanroom*, Programação Extrema *Extreme Programming* (XP), *Scrum* e FDD , destaca-se a metodologia *Rational Unified Process* (RUP), criada pela *Rational Software Corporation*, adquirida pela empresa *International Business Machine* (IBM). Tal metodologia é amplamente utilizada em conjunto com a ferramenta CASE *Rational Rose* da mesma empresa (JACOBSON, BOOCH E RUMBAUGH, 2000). Esta última ferramenta, além de auxiliar a concepção do *software*, conforme a metodologia RUP, é capaz de gerar o código inicial ao final da fase de concepção, mais especificamente na fase de projeto, conforme Figura 16, para que os trabalhadores desenvolvedores possam, a partir desses, iniciar o processo de codificação do que fora definido na fase conceitual. Embora seja amplamente mutável, a metodologia RUP é considerada “pesada”, devido a sua utilização ser preferencialmente aplicável a grandes equipes de desenvolvimento em grandes projetos. Para a gerência do projeto, o RUP contempla uma solução disciplinada de como delegar e controlar tarefas e responsabilidades durante o processo de desenvolvimento de *software*. O RUP é, por si só, um produto de *software*. É modular e eletrônico e toda a sua metodologia é apoiada por diversas ferramentas de desenvolvimento integradas, tal como a ferramenta CASE *Rational Rose*.

Para conceber um projeto de engenharia é necessária a utilização de ferramentas de abstração³⁰, tal como as ferramentas matemáticas para abstrações geométricas. Conforme Teles (2005, p. 21)

um arquiteto, por exemplo, tem a possibilidade de utilizar uma planta baixa que o ajuda, bem como ajuda seu cliente a avaliar espaços, fluxos de trânsito, disposição de elementos, entre outros. Com ela, torna-se simples identificar contradições e omissões. Ao capturar a realidade geométrica, em uma abstração geométrica correspondente, o arquiteto tem à sua disposição uma ferramenta que facilita e melhora o seu trabalho.

Tal como um escritório de arquitetura ou uma fábrica de automóveis, a fase de concepção de *software* se assemelha ao trabalho do arquiteto ou engenheiro quando estes projetam novos modelos de produto, enquanto que a codificação durante a fase de execução do *software* se assemelha ao trabalho dos engenheiros no chão de fábrica. Entretanto, Kruchten (2001) compara o trabalho do trabalhador de *software*, tanto na fase de concepção quanto na fase de execução, ao esforço do engenheiro durante a fase de concepção de um novo produto de engenharia, devido ao fato da automação do processo de desenvolvimento de *software*, com a utilização de ferramentas CASE, não exercer um efeito tão significativo, quanto à velocidade de produção, quanto exercem na produção industrial.

As ferramentas CASE agregam um conjunto de outras ferramentas que, seguindo uma determinada metodologia de engenharia de *software*, auxiliam o trabalhador na concepção de sistemas complexos. Com o auxílio da ferramenta, esse trabalhador modela

³⁰ “Abstração é o processo ou resultado de generalização por redução do conteúdo da informação de fenômeno observável, normalmente para reter apenas a informação que é relevante para um propósito particular. Por exemplo, abstraindo uma bola de futebol de couro por uma bola de futebol retemos apenas a informação das propriedades e comportamento da bola genérica. Similarmente, abstraindo a felicidade para um estado emocional reduzi a quantidade de informação coberta pelo estado emocional. Cientistas da computação usam a abstração para entender e resolver problemas e comunicar suas soluções para o computador usando alguma linguagem computacional em particular” (LEWIS, 2006, 211-231).

os requisitos do produto de *software*, codificando tais requisitos a partir de outras ferramentas específicas que permitem a codificação do produto a partir de linguagens computacionais. Dessa forma, as ferramentas CASE estão para a concepção do produto de *software* da mesma forma que as linguagens computacionais estão para a codificação do produto durante a fase de execução. Assim, durante todo o processo de produção de *software*, a idéia inicial do produto é trabalhada por diversos perfis de trabalhadores, em diferentes ofícios, a fim de se chegar ao produto final. Embora no desenvolvimento de produtos de *software* menores e menos complexos o mesmo trabalhador atue durante as fases de concepção e execução, o perfil dos trabalhadores em ambas as fases tende a ser diferenciado. Na primeira, é requerido um perfil mais especializado, devido ao intenso foco no processo criativo. Nessa fase, o conjunto de ferramentas CASE auxiliam os trabalhadores no processo criativo, em que predomina um caráter de manufatura heterogênea, no sentido de que são empreendidas diversas interações entre trabalhadores e ferramentas até a obtenção do produto parcial ao final da fase, assim como a existência de ferramentas com diferentes graus de automatização. Embora a engenharia de *software* busque uma padronização da concepção de *software* e um enquadramento da forma de atuação do trabalhador, durante essa fase, o ritmo do trabalho é ainda ditado pelo próprio trabalhador, que a partir de sucessivas interações com as ferramentas, em diferentes ofícios ligados à fase de concepção, produz um produto parcial ao final da fase. Segundo Pressman, a concepção é conduzida por trabalhadores com ofício de analista de negócio para a definição do problema (conforme Figura 16) e de analista funcional para o detalhamento técnico do projeto. Partindo do pressuposto que o produto desenvolvido durante o processo de produção de *software* é uma seqüência de instruções executáveis que tem como finalidade a resolução de um problema ou necessidade oriunda do mundo material (TANENBAUM, 2001), o analista de negócio estuda problemas relacionados

com tal necessidade e elabora-os para encontrar soluções a partir do desenvolvimento de produtos de *software*. Em algumas empresas, é atribuído o ofício de analista de sistemas³¹, analista dos sistemas empresariais ou analista funcional, mas em outras os títulos têm posições e responsabilidades diferentes. Um analista de negócio trabalha para construir uma ponte entre o demandante ou cliente do produto de *software* e os demais membros do departamento de tecnologia da informação ou outros provedores de serviços externos, a fim de tratar o problema ou necessidade. Nesse sentido, recolhe e documenta as exigências do demandante convertendo-as em linguagem pré-codificada, com base no modelo de engenharia de *software* utilizado, para que, a partir dessa modelagem auxiliada pelas ferramentas CASE, os analistas funcionais possam refinar tais requisitos de forma que estejam prontos para serem codificados durante a fase de execução ou codificação. O analista de negócio interpreta termos do projeto e esclarece as exigências do negócio a partir do demandante, visando estabelecer um sistema de comunicação eficaz para todo o projeto a partir de uma definição de escopo mais claro possível. Nesse sentido, recolhe dados e analisa a informação, envolvendo a equipe de funcionários do cliente e os usuários finais do produto de *software* (SUTJNI, 2007). As ferramentas CASE possibilitam o desenvolvimento desses *softwares*, sendo que tais ferramentas aprisionam uma série de habilidades (*skills*) dos trabalhadores da fase de concepção. Da mesma forma que o trabalhador do chão de fábrica (*shop floor*) tem as suas habilidades congeladas em sistemas de manufatura flexível, tendo as suas ações produtivas sido decompostas nas formas fundamentais do movimento e recompostas em operações mecânicas transferíveis

³¹ O desenvolvimento do *software* tem um processo genérico que se divide em fases: análise do sistema, projeto, desenvolvimento e implantação. Até a década de 70, o desenvolvimento era de responsabilidade dos programadores e a análise e projeto, considerados mais nobres e como profissões de nível superior, era de responsabilidade do analista de sistemas. Desde o início da década de 80, essa distinção diminuiu e muitos analistas de sistema são responsáveis por todas as fases do processo. Estamos, portanto, em uma situação em que o nome da profissão, "analista de sistemas", deriva apenas de uma das fases do processo de desenvolvimento de *software*, que é importante, mas não mais que as demais e o nome de cursos acadêmicos, análise de sistemas, reflete a profissão de um modo limitado, assim como os demais nomes atribuídos à profissão – projetista de *software*, engenheiro de *software* – são nomes baseados em processos (MASIERO, 1999).

às máquinas, também os trabalhadores conceituais ou engenheiros de *software* têm o seu conhecimento codificado, porém sob a forma de *software*.

À medida que a tecnologia de automação flexível evolui, é possível observar o caráter transitório das qualificações exigidas dos trabalhadores. Embora conhecimentos específicos de matemática e raciocínio lógico sejam requeridos do engenheiro de *software* no uso de ferramentas CASE para concepção e posterior codificação dos sistemas, ressaltando dessa forma a importância do trabalhador no processo produtivo, as rotinas de programação, aliadas ao suporte dessas ferramentas de concepção, permitem um processo de codificação e padronização do conhecimento, ou a busca incessante deste, do início (concepção) ao fim do processo. A contínua utilização das ferramentas CASE no processo de concepção de *software*, decompõe gradualmente formas fundamentais de conhecimento que são progressivamente incorporadas na otimização das próprias ferramentas CASE, gerando uma intelectualização de todo o processo de trabalho. Teles (2005, p. 52) assegura que

em princípio, o RUP [por exemplo] estabelece o desenvolvimento iterativo e incremental como forma de incorporar feedback e aprendizado ao processo de desenvolvimento. Entretanto, é comum equipes adotarem o RUP com iterações muito longas ou simplesmente executarem o projeto inteiro em uma única iteração. Nestes casos, o que se observa é a equipe executando um projeto de acordo com o processo em cascata, mas basicamente usando os artefatos³² do RUP para organizar a documentação. [...] A evolução sequencial do projeto se assemelha a uma fábrica onde requisitos são tratados como matérias primas que são transformadas à medida que avançam pela linha de produção. Cada transformação gera um conjunto de artefatos a serem utilizados em etapas posteriores da fabricação. Procura-se assegurar que cada artefato seja produzido corretamente para que o resultado final possa ser alcançado com a mesma previsibilidade e determinismo de uma fábrica. Portanto, evitar variações é fundamental.

³² Artefatos são produtos de trabalho finais ou intermediários produzidos e usados durante os projetos. Os artefatos são usados para capturar e transmitir informações do projeto. Um artefato pode ser um dos seguintes elementos: um documento, como no caso de negócio, ou documento de arquitetura de *software*, um modelo, como o modelo de casos de uso ou o modelo de *design*, ou um elemento do modelo, ou seja, um elemento existente em um modelo, como uma classe ou um subsistema (RATIONAL, 2001).

Com o auxílio das ferramentas de concepção de *software*, é possível o desenvolvimento de sistemas CAD/CAM, assim como os demais sistemas que integram máquinas industriais através da automação desses dispositivos, a partir da utilização de unidades de controle numérico e do desenvolvimento de interfaces de *software* que facilitem essa integração. A tarefa de integração dos vários dispositivos de um sistema de manufatura apresenta requisitos como: suporte de *hardware* (conexão física dos componentes) e suporte de *software* (conexão lógica dos componentes) adequados. Além disso, é necessário um *software* adicional para coordenar as operações dos diversos dispositivos e para implementar as estratégias de controle desses sistemas integrados. Esse *software*, usualmente, recebe o nome de *software* de controle (FRIEDRICH, 1996). Em tal processo de trabalho ocorre a intensificação do tratamento massificado das informações envolvidas, quando a concepção do produto, a partir do *software*, está diretamente ligada à peça, permitindo a programação das máquinas no chão-de-fábrica, ao mesmo tempo em que potencializa a flexibilidade do processo produtivo no momento em que permite a reprogramação das máquinas no chão de fábrica. Dessa forma, como aponta Tauile (1984, p. 128), o processo de produção de *software* se caracteriza como um processo extremamente intelectualizado, pois

é curioso apontar uma vez mais o caráter autofágico do trabalho social que gera esta transitoriedade. Ao programar, o programador está criando condições objetivas para que suas atividades em si sejam também programadas. Ao programar está programando a própria programação.

Podemos dizer, portanto, que a subordinação do trabalho ao capital no processo de desenvolvimento das ferramentas de concepção de *software* e a incorporação do conhecimento do engenheiro de *software* à ferramenta CASE para o desenvolvimento de *software*, encontram-se em um momento histórico de “acumulação primitiva do

conhecimento” (BOLAÑO, 2000), similar ao ocorrido no período manufatureiro prévio à Primeira Revolução Industrial.

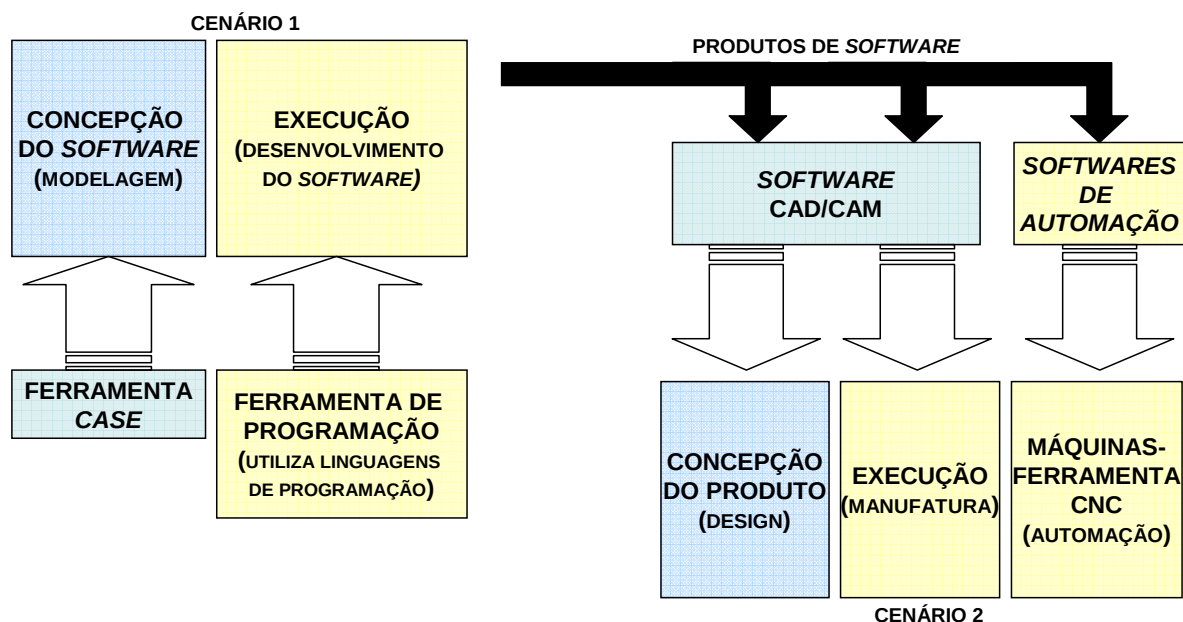


Figura 17 – processo de desenvolvimento de *software* (ferramentas) CAD-CAM para utilização em máquinas automatizadas.

Na Figura 17, concebida a partir do processo de desenvolvimento de *software* descrito em SEI (2007) e do processo de produção industrial automatizado com uso de ferramentas de *software* CAD/CAM descrito por Castelltort (1988) e Feldens (2000) , é possível analisar a codificação do conhecimento a partir de artefatos computacionais que assumem papéis distintos durante os dois cenários de produção: a concepção e execução do *software* e o seu posterior uso na indústria. O primeiro cenário é caracterizado pelo trabalho não-industrial intelectualizado e o segundo é parte fundamental de mediação e integração da automatização do trabalho industrial. Ambos os cenários são divididos basicamente nas fases de concepção e execução. No primeiro, artefatos de *software* compostos por ferramentas computacionais baseadas em metodologias específicas, permitem a elaboração (concepção) e produção (execução) de produtos de *software*. No

segundo cenário, os produtos elaborados durante o processo de produção de *software* são usados como ferramentas que integrarão sistemas compostos por *hardware* (máquinas automatizadas) e *software*, que permitem a concepção assistida por computador. Esses *softwares*, além de consistirem em formas automatizadas de concepção de novos produtos, permitem, ao final da fase de concepção, o fornecimento de todas as informações necessárias às máquinas automatizadas para que estas sejam capazes de produzir e elaborar o produto concebido.

Conforme o processo de *software* que compõe o primeiro cenário apresentado na Figura 17, o produto de *software* é normalmente produzido em um ambiente chamado de **fábrica de *software***. A expressão tem sido usada desde os anos 60, nos Estados Unidos, e anos 70, no Japão, conforme SWANSON (1991). No contexto do *software*, a analogia com a fábrica pode ser aplicada apenas aos objetivos da produção baseada no estilo industrial e não na sua implementação. A manufatura de *software* envolve pouca ou nenhuma produção tradicional: todo sistema é único; apenas partes individuais podem aparecer repetidamente em mais de um sistema. Uma importante característica de uma fábrica de *software* é a importância que ela dá à informação acumulada de vários projetos. Essa informação pode adquirir várias formas, incluindo elementos reutilizáveis (código, projeto e documentação), medidas de *performance*, relatórios da aplicação de técnicas específicas e processos de desenvolvimento (FERNSTRÖM 1992). Segundo Fernandes (2007), a fábrica de *software* constitui um processo estruturado, controlado e melhorado de forma contínua, considerando abordagens de engenharia industrial, orientado para atendimento a múltiplas demandas de natureza e escopo distintas, visando à geração de produtos de *software*, conforme os requerimentos documentados dos usuários ou demandantes de *software*, da forma mais produtiva e econômica possível. No século XX o

conceito foi bastante difundido, com muitas empresas de prestação de serviço com fábricas de *software* implantadas, inclusive aquelas em que a finalidade principal não é o desenvolvimento de produtos de *software*. Em outras empresas, a conotação de fábrica de *software* está mais ligada ao serviço de fábrica de programas, que se dedica exclusivamente à codificação de programas. Nesse sentido, uma fábrica de *software* pode possuir escopos distintos de atuação, desde a concepção do produto de *software*, até exclusivamente a codificação de programas de computador (FERNANDES, 2007), tal como demonstrado na Figura 18 – Escopo de atuação da fábrica de software (FERNANDES, 2007). em **fábrica de projetos**, projetos de *software* e projetos físicos.

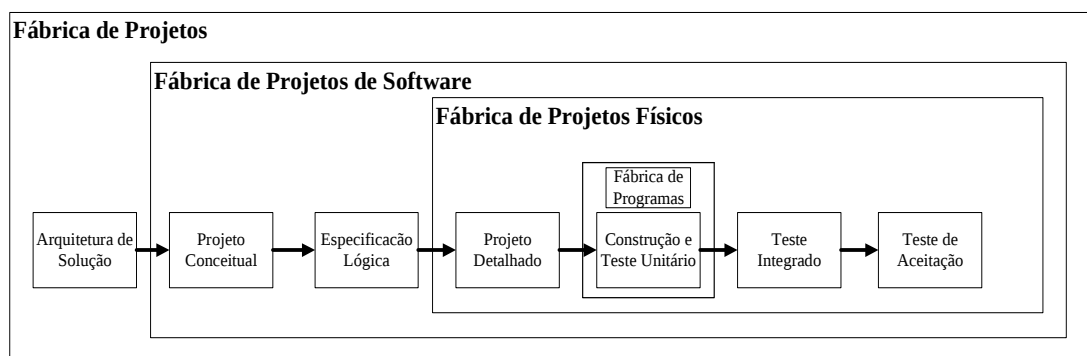


Figura 18 – Escopo de atuação da fábrica de software (FERNANDES, 2007).

Conforme Fernandes, a fábrica de projetos se destina à arquitetura da solução ou resolução de uma problemática em que uma parte é passível de ser solucionada por intermédio de um produto de *software*. A fábrica de projetos abrange o estágio que antecede a concepção do produto de *software* e se destina a projetar uma solução, em que o *software* é apenas um dos componentes, vislumbrando a implantação de processos e rotinas de operações de trabalho compostas por *hardware*, serviços e equipamentos de rede e de telecomunicações. O objetivo da fábrica de programas é executar (codificar) as ações planejadas durante a fase de concepção, considerando seus respectivos critérios de

elaboração e prazos conforme definidos previamente. A complexidade, profundidade de integração e interdependência dos diversos elementos da fábrica de *software* estão diretamente ligados ao seu escopo de atuação. Tal qual apresentado na Figura 19 – Escopo de atuação da fábrica de *software* conforme a complexidade dos processos de produção (FERNANDES, 2007). O controle da fábrica de projetos é bem mais complexo do que o de um escopo de uma fábrica de programas, por exemplo.

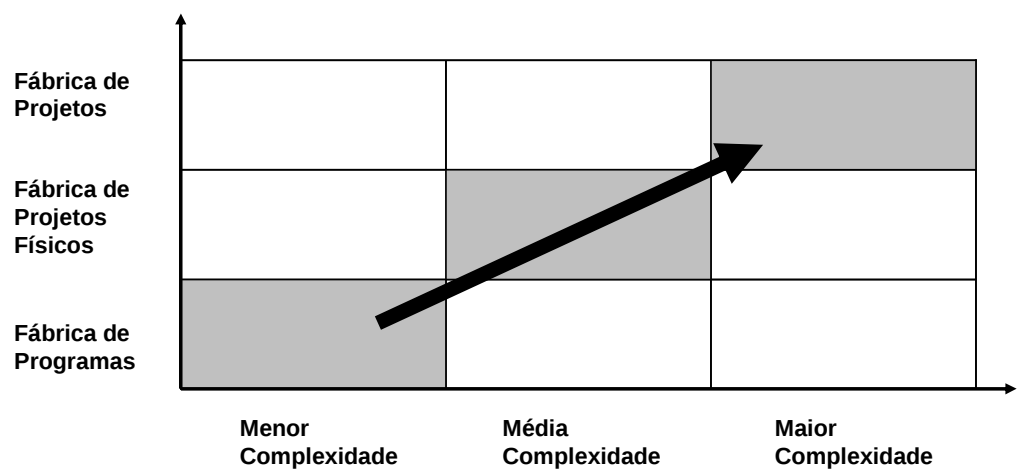


Figura 19 – Escopo de atuação da fábrica de *software* conforme a complexidade dos processos de produção (FERNANDES, 2007).

Devido aos *softwares* serem constituídos de partes menores ou módulos que podem ser construídos e acoplados até a montagem do produto final, as partes individuais podem ser desenvolvidas independentes uma das outras, desde que planejadas considerando nesse processo de trabalho um alto grau de complexidade e controle (Oliveira e Neto, 2003). Conforme os diversas demandas que podem ser assumidas pela fábrica de *software*, além do grau de complexidade e modularidade dos componentes, a forma de organizar a produção pode ser estabelecida de maneiras distintas, a partir de células de produção e geralmente condicionada pela linha de serviço ou mais precisamente pela linguagem de programação. O arranjo de produção pode ainda estar

geograficamente distribuído em um ou mais células de produção e a gestão da operação pode estar centralizada em relação a todas as células. Dessa forma, o processo de produção ocorre no nível de cada local e as etapas de concepção e execução se distinguem acentuadamente, tanto em relação ao grau de importância que assumem no processo quanto geograficamente.

Durante a etapa de concepção ou modelagem, o *software* é concebido a partir das ferramentas de concepção (CASE) por intermédio de trabalhadores especializados (engenheiros de *software*). O processo de concepção de tais sistemas culmina com a geração, automaticamente, pela ferramenta CASE, do código estruturado, pronto para ser completado na fase seguinte. A ferramenta CASE, através de um ambiente gráfico e virtual provido de diagramas lógicos, oferece aos engenheiros de *software* todas as facilidades para a concepção do sistema a ser codificado, produzindo como resultado, o código inicial (ou diagramas em alguns casos) pronto para ser detalhado pelo outro grupo de trabalhadores (programadores) durante a fase de execução ou desenvolvimento. A partir daí, os diversos pedaços (módulos³³) do sistema são distribuídos entre os trabalhadores (programadores) da fábrica de *software*, que se ocupam em codificar as especificações oriundas da fase de concepção, a partir dos diagramas ou código inicial gerado pelas ferramentas CASE, seguindo padrões e normas de codificação (programação) previamente estabelecidos. Durante o processo de codificação de *software*, vários trechos de código são reutilizados, devido à constante repetição e rotinização da

³³ Um módulo é uma parte do sistema que deve desempenhar uma função específica, de maneira que o seu funcionamento deva estar oculto a outros módulos e dedicado a uma função específica, assim como a interface de um módulo deve precisar somente das informações necessárias para que o módulo complete a tarefa (RUMBAUGH, 2006).

programação, culminando em trechos de código que são completamente padronizados, criando assim uma biblioteca de rotinas e padrões de solução de código³⁴.

Em cada uma das fases (concepção e desenvolvimento de *software*), os papéis desempenhados pelos artefatos de *software*, ferramentas CASE e ferramentas de programação são distintos. Sua interação, como ferramenta, com o trabalhador revela o grau de incorporação do conhecimento desse trabalhador no processo de valorização do capital. Durante a fase de concepção, as ferramentas de modelagem de *software*, tais como as ferramentas CASE, assumem o papel de auxiliar o trabalhador na concepção do produto, nesse caso um produto de *software*. Tal processo de concepção, para a criação de um produto de *software* pelos trabalhadores é diferenciado do processo de concepção do produto na indústria, quando auxiliado pela ferramenta CAD. Segundo Tauile (1984), o trabalho manual pode ser qualificado (TMQ) ou desqualificado (TMD). Da mesma forma, o trabalho intelectual pode ser de concepção criativa (TICC) ou de execução de rotinas pré-programáveis (TIERPP). De acordo com essa classificação, a concepção de um produto de *software* se enquadra em um trabalho intelectual de concepção criativa, mesmo que auxiliado pela ferramenta CASE, diferentemente do processo de concepção de produto industrial auxiliado pela ferramenta CAD, que se enquadra na classificação de trabalho intelectual de execução de rotinas pré-programáveis. O processo de produção de *software* gera um produto, como por exemplo a ferramenta CAD, que será utilizado na fase de concepção de produtos industriais, permitindo aos trabalhadores industriais envolvidos na fase de concepção executarem rotinas pré-programáveis sob a forma de ferramentas de *software* (ferramentas CAD). No ambiente da fábrica de *software*, onde

³⁴ Reuso é o processo de implementação e atualização de sistemas de *software* utilizando-se *software* e trechos de códigos pré-existentes. O reuso de *software* geralmente ocorre entre sistemas similares ou entre sistemas não similares de forma genérica (GERARD, 2008).

todo o processo de produção de *software* é intelectualizado do início ao fim, os indivíduos atuantes no processo estão agrupados nas fases de concepção e execução. Na primeira fase, tratando-se de concepção criativa e, na segunda, trabalho intelectual de execução de rotinas pré-programáveis.

No processo de desenvolvimento de *software*, tal como conduzido em uma fábrica de *software*, devem ser considerados adicionalmente dois fatores importantes. O primeiro consiste nos diferentes tipos de linguagem de programação. Tal fator é estabelecido conforme diferentes arquiteturas de computadores e diferentes finalidades de programação. Em uma mesma arquitetura coexistem diversos tipos de linguagem de programação, com diferentes finalidades, conforme as características do sistema a ser produzido. Algumas linguagens, por exemplo, são voltadas para o desenvolvimento de *softwares* comerciais, com ênfase em controles financeiros, contábeis e gerência de estoques. Outros são voltados para aplicações mais específicas, como *softwares* de manipulação de imagens e vídeos. Nos primeiros, as linguagens de programação estão voltadas para prover facilidades de armazenamento, organização e acesso a bases de dados, assim como prover facilidades no desenvolvimento de interfaces de uso amigáveis. No segundo caso, as linguagens permitem maior facilidade de manipulação e processamento de cálculos complexos na memória do computador com o intuito de simplificar a execução, com o maior grau de consistência possível dos dados.

O segundo fator importante a ser considerado são os modelos de referência para o desenvolvimento e controle da organização do trabalho de desenvolvimento de *software*,

que se propõem a eliminar as inconsistências, aumentar a clareza e entendimento, estabelecer terminologia comum e definir regras uniformes³⁵.

Os modelos de referência estabelecem uma padronização capaz de definir as regras segundo as quais o trabalho de desenvolvimento de *software* deve estar organizado, de maneira a prover os meios de controle desse trabalho intelectual. Tais modelos de referência tentam definir um maior controle sobre os meios de produção, nesse caso o trabalho intelectual do programador e engenheiro de *software*, no sentido de que a fábrica de *software* não possui conhecimento específico em nenhuma área de negócio que os sistemas produzidos se propõem a informatizar. Considerando como uma área de atuação, o conhecimento específico sobre determinada regra implementada por um *software*, a fábrica de *software* não precisa necessariamente possuir o conhecimento prévio de tal regra para que possa, por intermédio dos seus engenheiros de *software*, conceber e produzir um produto de *software* em uma determinada área de atuação. Ou seja, a fábrica de *software* não precisa possuir o conhecimento acumulado sobre procedimentos médicos para que possa produzir um *software* de diagnóstico clínico capaz de satisfazer as necessidades de uma determinada área de atuação da medicina.

As forças naturais presentes no processo de produção de *software* se constituem no conhecimento do trabalhador durante o processo de produção. Esse conhecimento se constitui como elemento fundamental para, a partir dos artefatos de *software* e ferramentas CASE, a transformação do conhecimento em produto de *software*. Para apropriar-se do

³⁵ O CMMI (Capability Maturity Model Integration) é um modelo de referência que fornece orientação para o desenvolvimento de processos de *softwares* e tem como objetivos eliminar suas inconsistências; aumentar sua clareza e entendimento; fornecer uma terminologia comum e um estilo consistente; estabelecer regras de construção uniformes e assegurar consistência com a ISO/IEC 15504. O CMMI constitui tanto um modelo de capacidade como um modelo de maturidade. O modelo dentro de uma empresa pode ser alcançado em etapas consecutivas, representando a idéia de maturidade, ou também de maneira contínua, quando são mensuradas as capacidades em práticas individuais (SEI, 2007).

conhecimento dos trabalhadores no processo produtivo, a empresa necessita dos modelos de referência, artefatos de *software*, ferramentas de concepção e desenvolvimento padronizados. Nesse sentido, de acordo com Bolaño e Herscovici (2005, p. 11-12), ficam evidentes os principais aspectos da codificação do conhecimento do trabalhador intelectual e o controle total desse tipo de processo produtivo de trabalho:

A Economia distingue hoje normalmente dois tipos de Conhecimento: tácito e codificado; o primeiro é diretamente ligado ao trabalhador que o detém, não podendo ser utilizado sem a sua participação, enquanto o segundo é despersonalizado, descontextualizado e fixado sobre um suporte material: o livro, o CD, a rede. Capital variável, num caso, constante, no segundo. Já não se trata mais aqui de uma acumulação primitiva do conhecimento, como aquela realizada pelo capital no período manufatureiro, mas de uma dinâmica concorrencial de codificação do conhecimento tácito existente e, com isso, a criação de novo conhecimento tácito necessário para a interpretação dos códigos, das linguagens e a utilização estratégica do novo conhecimento produzido e assim sucessivamente.

O trabalhador (engenheiro de *software*) fica limitado intelectualmente a partir do momento em que deve enquadrar o seu pensamento no modelo produtivo de concepção do trabalho (ferramentas CASE) e modelos de referência como forma de garantir a execução da sua criação e o trabalhador (programador/reprogramador/operador) tem as suas tarefas de execução ampliadas, no sentido de assumir um papel triplo que lhe exige novos conhecimentos e habilidades.

Entretanto, os métodos e processos de desenvolvimento de *software* existentes não tratam de forma satisfatória a correspondência entre o ambiente organizacional e o sistema desenvolvido (D'SOUZA, 1998). De forma geral, as ferramentas CASE e modelos de referência fornecem técnicas para modelar aspectos organizacionais e de negócio, mas ainda não estabelecem uma clara conexão entre esses elementos e a especificação dos requisitos do sistema (KRUCHTEN, 2000). Um ponto relevante é o aprimoramento e desenvolvimento de recursos técnicos e humanos mais apropriados na

elucidação, documentação e validação de requisitos de *software*. Isto implica não somente em técnicas mais efetivas aplicadas em atividades no processo de engenharia de requisitos (GOGUEN, 1993), mas também em uma melhor formação multidisciplinar dos profissionais envolvidos. Dessa forma, objetiva-se propiciar aos engenheiros de requisitos, o entendimento da linguagem dos usuários do sistema e sua relação com as tarefas do trabalho das pessoas. Assim, familiarizado com o ambiente e linguagem dos usuários, o engenheiro de requisitos pode desenvolver documentos de requisitos com menos ambigüidades e incompreensões.

A partir de uma formação matemática baseada em raciocínio lógico, os engenheiros e programadores de *software* são treinados para resolver problemas apresentados sob a forma de demanda de *software* - utilizando-se para isso das ferramentas de programação e artefatos de *software* durante o processo de produção e desenvolvimento. Dessa forma, a empresa (fábrica de *software*), se encontra em uma condição sob a qual não agrega nenhum valor ao trabalho além do trabalho dos seus funcionários, pois não possui conhecimento específico sobre nenhuma especialidade em particular. Dessa forma, conforme Bolaño (2004b, p. 70), nota-se a importância renovada do trabalho com informação:

a Economia do Conhecimento, centrada na recorrente interação entre conhecimento tácito e codificado, depende hoje da existência de enormes bancos de dados informatizados, que deverão ser constantemente renovados, não simplesmente no sentido de que novos dados serão agregados, mas no de uma codificação ininterrupta, que exige o manejo de códigos em permanente atualização, articulando os trabalhadores intelectuais dos diferentes campos do saber, usuários daqueles bancos, e os trabalhadores da informação responsáveis pela sua manutenção como ambientes vivos de comunicação produtiva.

Existe também atualmente uma preocupação em relação ao nível de precisão adequado que deve ser utilizado em atividades na engenharia de requisitos, mais

especificamente para a delimitação do produto demandado à fábrica de *software*. Quando requisitos são explicitados, documentados e validados, podem ser utilizadas técnicas com maior ou menor precisão, aliadas a métodos formais, os quais possuem sintaxe e semântica precisas e passíveis de validação. Por outro lado, técnicas tradicionais são mais informais e os artefatos produzidos por essas técnicas são difíceis de serem verificados e validados com completa segurança e exatidão.

O perfil profissional que se configura a partir da introdução das MFCN, tal como verificado desde o início da maquinofatura, se consolida no perfil de operários altamente especializados e técnicos de processo, devido ao trabalho ser dividido em concepção e execução (operação da máquina), o que faz com que uma parte significativa do trabalho seja concentrado na programação, traduzindo-se pela intensificação do trabalho especializado e na ampliação do tempo dedicado a tarefas de escritório em relação ao tempo gasto anteriormente na indústria e ao trabalho no chão de fábrica.

ANÁLISES CONCLUSIVAS

A **linguagem computacional** é o meio pelo qual se torna possível o desenvolvimento de *softwares* a partir da codificação e controle da concepção (fase de *design* ou projeto) do sistema que se pretende desenvolver. Tal controle se concretiza no enquadramento da concepção planejada, direcionada pela engenharia de *software* e materializada nas linguagens de programação, e a execução objetivada, materializada nas máquinas. A linguagem de programação implica dois tipos distintos de subsunção. O primeiro tipo é caracterizado pelo enquadramento requerido pelo programador à gramática formal definida na linguagem de programação, assim como metodologias de desenvolvimento de *software*, para a concepção do programa de computador, em um processo de criação de *software* (tal como programas de computador que são executados nas máquinas automatizadas) por meio de *software* (como por exemplo, programas que permitem a programação em linguagens de alto nível, como Fortran, Cobol e Algol). Nesse tipo de produção, um *software* assume o papel de ferramenta de concepção e desenvolvimento de um produto que é também um *software*.

O segundo tipo de subsunção é decorrente do produto originado do primeiro tipo. Os *softwares* desenvolvidos para a indústria, tal como o CAD/CAM, permitem tanto a concepção planejada dos produtos industriais quanto a execução objetivadas operada pelas máquinas. Os *softwares* industriais desenvolvidos para serem executados nas máquinas automatizadas são passíveis de serem reprogramados no chão de fábrica através de interfaces simplificadas nos Controles Numéricos Computadorizados – CNC, expandindo dessa forma as atividades dos “operadores de máquina que só operam” (LEITE, 1986,

p.35). Tais operadores assumem também a função de programadores de máquina no chão de fábrica.

O avanço da microeletrônica intensificou tal ampliação com a criação de *softwares* computacionais que integram o processo produtivo desde a sua concepção (*design*) até a sua execução (*manufacturing*). As ferramentas CAD/CAM, segundo a taxonomia definida por Cainarca et al. (1989) são exemplos concretos de sistemas de *design* flexível (*flexible design systems*), os quais possibilitam que o conhecimento acumulado esteja codificado e disponível sob a forma de *software* com a finalidade de automatizar o processo de trabalho desde sua concepção até a execução final.

A passagem das MFU para as MFCN se caracteriza pela tendência de apagamento da dependência que as primeiras tinham de determinadas habilidades do trabalho vivo e pela incorporação de novas habilidades requeridas por estes últimos. Dessa forma, o trabalhador operador limita-se a manter o equipamento provido de matéria prima e vigiar o processo de trabalho mecanizado, enquanto o trabalhador programador exerce a tarefa de programar a máquina. Conforme o estudo realizado por Elenice Leite, observa-se que, embora existam trabalhadores especializados em programar as MFCN, essa tarefa se apresenta mais comumente como parte integrante das atividades do perfil de técnicos de processo, ocorrendo dessa forma uma homogeneização da base técnica, já que em alguns casos os próprios operadores tornam-se programadores das MFCN. No caso brasileiro, há cerca de uma década e meia, verificou-se que um quarto dos programadores havia sido anteriormente operador e um número significativo destes estudava engenharia ou era engenheiro (TAUILE, 1984). Outra característica do mesmo processo de trabalho é que os operadores de máquinas (técnicos de processo) perdem espaço devido à simplificação do

trabalho ocorrida por incorporação pelas MFCN de parte da atividade antes requerida do trabalho vivo. Em contrapartida, novas habilidades são requeridas dos trabalhadores remanescentes devido à complexidade na preparação da máquina, a partir da programação definida pelos programadores. Nesse sentido, o perfil de programador (que só programa) coexiste com o perfil operador/programador devido à familiaridade destes últimos com as atividades práticas de execução. Entretanto nem todos os operadores possuem habilidades para se tornarem operador/programador e observou-se, a partir do perfil dos profissionais das empresas avaliadas, no estudo de Elenice Leite citado, uma tendência das empresas de buscar profissionais mais completos, capazes de manejar os novos equipamentos com maior segurança, independência e responsabilidade.

De forma similar, a convergência de funções foi observada em um estudo realizado por Romeiro Filho (1996) a partir da implantação do CAD na Companhia Siderúrgica Nacional – CSN. No estudo, foi observada a associação do *software* CAD a uma “prancheta eletrônica” na qual os usuários deveriam necessariamente desenhar. Conforme Romeiro Filho (1996, p. 117),

no antigo processo de trabalho os projetos eram definidos pelo engenheiro, detalhados pelo projetista e executados pelo desenhista, sendo que os dois últimos interagiam para a definição dos elementos, materiais, resistências etc. Após o detalhamento do projeto, este era analisado pelo engenheiro, para a sua aprovação final. Com a introdução do CAD esta hierarquia foi rompida, já que a figura do desenhista (e não a do projetista) tornava-se desnecessária, o que causou uma certa confusão acerca das atribuições de cada profissional envolvido.

Conforme se observa, o projetista poderia desenhar diretamente na tela do computador, dispensando a ação do desenhista copista, caracterizando uma quebra de hierarquia conforme a estrutura organizacional vigente, na medida em que o projetista tende a executar uma tarefa menos nobre (de desenhista), embora, ao longo do seu

desenvolvimento profissional, o projetista tenha exercido a função de desenhista. Este pode ser considerado resultado da exclusão de um ofício do processo de trabalho, a partir da perda de autonomia do desenhista copista e da incorporação de parte do trabalho pelo projetista.

Devido ao acentuado desenvolvimento das máquinas, o ato de programar está inserido em um contexto de desenvolvimento e produção definido e pré-determinado, que extrapola o conceito de comunicação e interface entre homem e máquina, constituindo toda uma “forma de pensar” a serviço do capital produtivo, com o trabalhador excluído gradativamente do processo de trabalho que age de modo automatizado tornando o operário descartável. Essa característica foi observada por Moraes Neto (1991) ao verificar que dessa forma o capital se libertou gradativamente da dependência direta do trabalho vivo no processo de trabalho, sendo a maquinaria, a forma de produção responsável pela superação radical do ofício manual da manufatura, pelo revolucionamento do instrumento de trabalho, quando então o capital encontra a sua base técnica adequada. No processo produtivo envolvendo homem e máquina, ou operador e máquina CNC por exemplo, repousa todo o processo de subsunção do trabalho ocorrido desde a sua forma artesanal até a automação. O operador assume papel de coadjuvante no processo de trabalho por ter as suas habilidades manuais transferidas quase que integralmente para a máquina.

Por outro lado, na produção de *software* por meio de *software* existe uma participação acentuada do trabalho vivo (do programador de computador) em um processo de subsunção do trabalho intelectual inacabado, por conta da subjetividade envolvida na

tarefa de codificação de programas a partir de linguagens computacionais para a criação de *softwares*, que por sua vez poderão ser reprogramados no chão de fábrica.

Na indústria que faz uso de ferramentas CAD/CAM em um sistema de produção flexível, o trabalho do programador (preparador) das máquinas requer maior habilidade e especialização do que o trabalho dos operadores (supervisores de máquina). De maneira análoga, no processo de desenvolvimento de *software*, o trabalho do projetista de *software* requer maior habilidade do que a função de programador da ferramenta. No processo de desenvolvimento de sistemas, o *software* assume ao mesmo tempo o papel de ferramenta e produto, tal como na produção de máquinas por meio de máquinas apontada por Marx, em que o trabalhador deixa de exercer o papel central no processo de trabalho. A diferença no processo de produção de *software* em relação ao processo apontado por Marx reside no tipo do trabalho, que deixa de ser manual, passando a ser completamente intelectualizado. A produção de máquinas na grande indústria permitiu o avanço extremo da especialização e precisão dos movimentos exercidos por máquinas automatizadas, produzindo uma maior quantidade de produtos em menor tempo e com precisão apurada. No processo de produção industrial que faz uso de sistemas CAD/CAM, a intelectualização domina todo o processo, até a concepção da própria ferramenta utilizada na produção final.

BIBLIOGRAFIA

ANTUNES, Ricardo. **Adeus ao trabalho? Ensaio sobre as Metamorfoses e a Centralidade do Mundo do Trabalho**. São Paulo: Cortez, 1995.

AYRES, R. U., **Technology Forecast for CIM**. Manufacturing Review, Vol.2, No. 1, 1989, pp. 43-52.

BALDAM, Roquemar; COSTA, Lourenço. **AUTOCAD 2009 - Utilizando totalmente**. Editora Érica, 2009.

BOEHM, C. W. et al. Prototyping vs. Specification: A MultiProject Experiment. Proc. Of seventh Int. **Conf. On Software Engineering**, New York: ACM and IEEE, 1984.

BOLAÑO, C. R. S. (Org.) . **Os Processos de Globalização e Mundialização: Tecnologias, Estratégias e Conteúdo**. Anais do III Colóquio Brasil-França de Pesquisadores. Aracaju: Editora UFS, 1995.

BOLAÑO, César. **Indústria cultural, informação e capitalismo**. São Paulo: Hucitec, 2000.

BOLAÑO, C. R. S. Trabalho Intelectual, Comunicação e Capitalismo. A re-configuração do fator subjetivo na atual reestruturação produtiva. **Revista da Sociedade Brasileira de Economia Política** , Rio de Janeiro, 2002.

BOLAÑO, C. R. S. **A Economia Política da Internet e sua Crise**. In: Comunicação, Informação e Cultura - dinâmicas globais e estruturas de poder. 00 ed. Salvador: EDUFBA, 2004, v.01, p. 43-65.

BOLANO, C. R. S. Economia Política do Conhecimento e o Projeto Genoma Humano do Câncer de São Paulo. In: **Revista Eptic On-Line**. Vol. VI nº 3, p. 67-97, Set.-Dez., 2004.

BOLAÑO, César e HERSCOVICI , Alain. A Crítica da Economia Política da Informação e do Conhecimento. **Revista de Economía Política de las Tecnologías de la Información y Comunicación**. 2005.

BONATTI, Ivani e MADUREIRA Marcos. **Introdução à Análise e Síntese de Circuitos Lógicos**, São Paulo: UNICAMP, 1990.

BOOCH, G; RUMBAUGH, J e JACOBSON, I: UML, Guia do Usuário: tradução; Fábio Freitas da Silva, Rio de Janeiro, Campus ,2000.

BOOLE, G; (1854). **An Investigation of the Laws of Thought: On Which Are Founded The Mathematical Theories Of Logic And Probabilities**. Kessinger Publishing.

BRAVERMAN, Harry. **Trabalho e Capital Monopolista – A degradação do trabalho no século XX**. Rio de Janeiro, Guanabara, 1974.

BROOKS, Frederick P. **No Silver Bullet —Essence and Accident in Software Engineering**. University of North Carolina at Chapel Hill, 1986.

CAINARCA, Gian Carlo, COLOMBO, Massimo e MARIOTTI, Sergio. **An evolutionary pattern of innovation diffusion. The case of flexible automation.** Research Policy, 1989, vol. 18, issue 2, pages 59-86.

CASTELLTORT, Xavier. **CAD/CAM: metodologias e aplicações práticas.** São Paulo. McGraw-Hill, 1988.

CUNHA, José da et al. **Computação gráfica e suas aplicações em CAD.** São Paulo. Atlas, 1987.

FAGGIN, F., KLEIN, T., e VADASZ, L. : **Insulated Gate Field Effect Transistor Integrated Circuits with Silicon Gates**, apresentado na reunião de IEEE International Electron Device. Washington, D.C., outubro 1968.

FELDENS, Luis Felipe. **Notas de Aula: Sistemas Integrados de Fabricação - Informatização Industrial CAD/CAM.** Disciplina de Sistemas de Fabricação. Pontifícia Universidade Católica do Rio Grande do Sul, PUC RS, Brasil. 2000.

FERNANDES, Aguinaldo Aragon. **Fábrica de software: implantação e gestão de perações.** São Paulo, Atlas 2007.

FERNSTRÖM, C., NÄRFELT, K., OHLSSON, L. **Software Factory Principles, Architecture, and Experiments.** IEEE Software, 1992.

FERRARI, Alysson. **Linguagens de Programação - História da Computação**, 2008. Disponível em <<http://fma.if.usp.br/~alysson>>. Acesso em 09/02/2009.

FRIEDRICH, Luis Fernando. **Uma Abordagem Distribuída no Desenvolvimento e Implementação do Software de Controle de Chão-de-Fábrica em Sistemas de Manufatura Celular.** Universidade Federal de Santa Catarina. Florianópolis, Fevereiro de 1996.

GOGUEN, Joseph A; LINDE, Charlotte; **Techniques for requirements elicitation.** In Proceedings of the IEEE International Symposium on Requirements Engineering- RE'93 San Diego-USA, IEEE Computer Society Press, 1993.

GONICK, Larry. **Introdução Ilustrada à Computação.** Editora Harbra Ltda, ed.1986.

INTERNET / MICROINFORMÁTICA. **História do computador e da Internet: 1800-1899.** Disponível em: <<http://ufpa.br/dicas/net1/int-h180.htm>>. Acesso em 20/04/2008.

J L HENNESSY, D A PATTERSON. **Organização e Projeto de Computadores - A Interface Hardware/Software.** 3a edição, Editora Campus, 2005.

KAPLINSKY, R. **Microelectronics and technical change revisited.** IDS/University of Sussex, march 1985.

KRUCHTEN, Phillipe. **What Is The Rational Unified Process.** The Rational Edge, IBM, 2001.

LABORATÓRIO DE USINAGEM & AUTOMAÇÃO. **CAD/CAM: Sistemas Integrados de Produção Visando Prototipagem Rápida.** Disponível em: <<http://www.demec.ufmg.br/Grupos/Usinagem/CADCAM.htm>>. Acesso em 31/09/2009.

LÉVÉNEZ, Éric. **Computer Languages History Timeline.** Disponível em <<http://www.levenez.com>>. Acesso em 08/10/2007.

LEWIS, James W. **Cortical Networks Related to Human Use of Tools.** Departamento de fisiologia e farmacologia e centro de imagens avançadas. West Virginia. 2006.

MACHADO, Aryoldo. **Comando Numérico Aplicado Às Máquinas.** Ícone. São Paulo, 1989.

TELES, Vinícius. **Um estudo de caso da adoção das práticas e valores do extreme programming.** Rio de Janeiro: UFRJ / IM / DCC, 2005.

MARTIN, M.B & KOCH, M. (1999) – **Escola e Computador.** Florianópolis, 1999. Monografia - Pedagogia, na matéria Pesquisa e Prática Pedagógica IV). Universidade Federal de Santa.

MARX, K. **O Capital: Crítica da economia política.** Rio de Janeiro: Civilização Brasileira, 1980.

MARX, K. **O Capital: Crítica da economia política.** São Paulo: Nova Cultural, 1985.

MASIERO, Paulo C. **Sistemas de informação ou análise de sistemas?** Disponível em <<http://www.icmc.sc.usp.br/~masiero/nomesist.htm>>. Acesso em 05/04/2009.

MENEZES, Romeu. **Material Básico Tecnologia da Informação.** Disponível em <www.ensino.rumix.com.br>. Acesso em 08/04/2007.

MORAES NETO, B.R. **Marx, Taylor, Ford as forças produtivas em discussão.** Brasiliense. São Paulo. 1991.

MORAES NETO, B.R. **Fordismo e ohnoísmo: trabalho e tecnologia na produção em massa.** Estudos Econômicos, vol 28, n.2, abril-junho 1998, item 1: Fordismo: trabalho e tecnologia.

MORAES NETO, B.R. **Século XX e trabalho industrial: taylorismo/ Fordismo, ohnoísmo e automação em debate.** São Paulo. Xamã. 2003.

NOBLE, D. **Social Choice in Machine Design In: A. Zimbalist (org). Case Studies on the labor process.** Nova Iorque: Monthly Review Press, 1989.

PACITTI, Tercio. **Construindo o futuro através da educação: do Fortran a internet.** São Paulo. Makron Books, 3ª edição, 1998.

PAPERT, Seymour. **LOGO: computadores e educação.** São Paulo: Brasiliense, 1985.

PINEDA, José Octávio de Carvalho. **A entropia segundo claudes shannon: O desenvolvimento do conceito fundamental da teoria da informação**. Pontifícia Universidade Católica. São Paulo, 2006.

PIROPO, Benito. **Analytical Engine II: A Máquina Diferencial**, 2006. Disponível em <<http://www.forumpcs.com.br/coluna.php?b=180093>>. Acesso em 06/11/2009.

RATIONAL, Software Corporation. **Rational Unified Process**, 2001. Disponível em: <<http://www.wthreex.com/rup/portugues/index.htm>>. Acesso em 26/11/2009.

RATTNER, Henrique. **Impactos sociais da automação: o caso do Japão**. São Paulo, Nobel, 1988.

ROMEIRO FILHO, Eduardo. **CAD na Indústria: Implantação e Gerenciamento**. Rio de Janeiro: Editora UFRJ, 1996.

RUMBAUGH, James. **Modelagem e Projetos baseados em objetos com uml2**. São Paulo: Elsevier, 2006.

S.L. PFLEEGER, **Engenharia de Software: Teoria e Prática**. São Paulo: Prentice Hall, 2ª edição, 2004.

SHANNON, Claude. **A Mathematical Theory of Communication**. Reprinted with corrections from The Bell System Technical Journal, Vol. 27, pp. 379–423, 623–656, July, October, 1948. Disponível em <<http://www.mast.queensu.ca/~math474/shannon1948.pdf>>. Acesso em 06/11/2009.

SIMHON, M. **Princípios dos sistemas flexíveis**. Máquinas e Metais, São Paulo. 1986

SMITH, Adam. **Riqueza das nações**. São Paulo: Abril Cultural, 1983 (Os Economistas).

SOHN-RETHEL, A. (1989). **Trabalho Espiritual e Corporal Para a Epistemologia da História Ocidental**. Rev. u. erg. Neuauflage. Weinheim, VCH, Acta Humaniora, 1989.

SUTJINI, H. **Trabalhos do analista do negócio**. Disponível em <<http://www.articlegarden.com/pt/article.php?id=41123&act=print>>. Acesso em 28/05/09.

SWANSON, Kent; MCCOMB, Dave; SMITH, Jill; McCUBBREY, Don. **The application software factory: applying total quality techniques to systems development**. MIS Quarterly, 1991.

TANENBAUM, Andrew S. **Organização Estruturada de Computadores**, Editora LTC, 4º ed. 2001.

TAUILE, José Ricardo. **Microeletrônica, Automação e desenvolvimento econômico**. Nova York, 1984. Tese de Doutorado, New School for social research.

TAUILE, José Ricardo. **Para (re)construir o Brasil contemporâneo: trabalho, tecnologia e acumulação**. Rio de Janeiro. 2001.

UFPA. **História do computador e da Internet: 1940-1949**. Disponível em <<http://ufpa.br/dicas/net1/int-h194.htm>>. Acesso em 20/04/09.

VIEIRA, Deodete Packer. **Facilitadores no processo de inovação tecnológica**. Departamento de Engenharia de Produção. Universidade Federal de Santa Catarina. 1995. Disponível em <<http://www.eps.ufsc.br/teses/deodete/indice/index.html>>. Acesso em 04/04/2008.

YOURDON, E., **Managing the System Life Cycle**. Englewood Cliffs, NJ, Yourdon Press, 1988.

ANEXOS

I - EVOLUÇÃO DAS LINGUAGENS DE PROGRAMAÇÃO

II - GLOSSÁRIO

1. Letras Inglesas

<i>Advanced Research Project Agency</i>	Agência de Pesquisas em Projetos Avançados.
<i>Assembler</i>	Linguagem simbólica muito próxima a linguagem de máquina
<i>Assembly</i>	Notação legível por humanos para o código de máquina que uma arquitetura de computador específica usa.
<i>Chip</i>	Dispositivo eletrônico o qual possui milhões de circuitos integrados.
<i>CIM - Computer-Integrated-Manufacturing</i>	Produção integrada por computador.
<i>Cleanroom</i>	Metodologia utilizada no desenvolvimento de <i>software</i> bastante difundida no desenvolvimento de grandes projetos corporativos.
<i>Design</i>	Processo técnico e criativo relacionado à configuração, concepção, elaboração e especificação de um artefato.
<i>DOD - Department of Defense</i>	Departamento de Defesa Norte Americano.
<i>Drivers</i>	Pequeno programa (<i>software</i>) que controla um equipamento (<i>hardware</i>) específico, como uma placa de componente eletrônico.
<i>Finite element modelling</i>	Modelagem de elementos finitos (técnica para a análise do comportamento de estruturas submetida a uma variedade de cargas)
<i>Flexible design systems</i>	Sistemas de elaboração flexível
<i>Hardware</i>	Equipamento
<i>Just in time</i>	Manutenção de matérias-primas e componentes em estoque apenas em quantidade suficiente para manter o processo produtivo no momento. No limite, as quantidades são nulas, pois os únicos estoques de matérias-primas existentes são os que estão precisamente a ser transformados (Ohno, 1997).
<i>Knowledge</i>	Conhecimento
<i>Manufacturing</i>	Manufatura ou fase de execução que sucede a fase de concepção
<i>Microchip(s)</i>	Pequeno material semicondutor que contém um transistor eletrônico ou circuito integrado
<i>National Science Foundation</i>	Fundação da ciência Nacional.
<i>Parsing</i>	Análise

<i>Performance</i>	Desempenho
<i>Shop-floor</i>	Chão de fábrica
<i>Skill</i>	Habilidade
<i>Slide-rest</i>	Descanso deslizante (torno)
<i>Software</i>	Programa de computador
<i>Software Houses</i>	Empresas que se dedicam ao desenvolvimento de programas de computador com fins comerciais.
<i>Transfer</i>	Transferência, máquina de réplica.
<i>Workstation</i>	Estação de trabalho, computador.

2. Conceitos

Manufatura heterogênea	Tipo de manufatura em que o artigo produzido se constitui pelo ajustamento mecânico de produtos parciais independentes (Marx, 1980, p. 397).
Manufatura orgânica	Tipo de manufatura em que o artigo produzido deve a sua forma acabada a uma seqüência de operações e manipulações conexas (Marx, 1980, p. 397).
Máquina diferencial	Invento de Charles Babbage, para cálculos com polinômios de forma automatizada.
Máquina analítica	Projeto de um computador mecânico moderno de uso geral, feito pelo professor de matemática britânico Charles Babbage.
Compilador	Programa de computador (<i>software</i>) que, a partir de um código fonte escrito em uma determinada linguagem de programação, cria um programa semanticamente equivalente, porém escrito em linguagem de máquina.
Fábrica de <i>software</i>	Conjunto de profissionais, materiais, processos e metodologias estruturadas de forma semelhante às das indústrias, utilizando práticas de processos de produção de <i>software</i> aperfeiçoados para o desenvolvimento, testes e manutenções dos <i>softwares</i> .
Cartões perfurados	Cartões destinados ao registro de dados por meio de perfuração. O cartão perfurado original, inventado pelo mecânico francês Joseph-Marie Jacquard, foi utilizado experimentalmente pelo departamento de saúde e higiene mental <i>New York City Board of Health</i> para fins estatísticos. Após este ensaio, os cartões perfurados foram adotados para uso no recenseamento americano de 1890 (Infopédia, 2009).
Computador eletromecânico	Computador que faz uso de relês.
Computador seqüencial	Um computador seqüencial é formado basicamente pelos seguintes elementos de hardware: um único processador, memória e dispositivos de entrada e saída. Um programa desenvolvido para esta arquitetura também é planejado para ser executado de forma seqüencial, e é totalmente dependente do número de instruções que o processador pode executar em seqüência, quanto mais instruções puderem ser executadas em um determinado período de tempo melhor é o processador. Porém a tecnologia empregada na construção de processadores é limitada (...a velocidade de operação dos circuitos eletrônicos não pode continuar a melhorar indefinidamente (Tanenbaum, 2001).
Linguagem de máquina	A linguagem mais básica do computador é a linguagem de máquina, aquela que o computador processa diretamente através dos circuitos do processador (pelo <i>hardware</i>).
Linguagem computacional	É a forma pelo qual o homem expressa as instruções a serem executadas pela máquina.