DESIGN OF PROGRAMS

Um programa é uma tecnologia, um sistema que, atuado por uma intenção, promove uma transição de outro sistema de um estado menos para um mais preferível. Existe então uma subjetividade inerente na definição, nessa “intenção” e “preferência”, sem a qual um programa não existe. Retirar a subjetividade, referindo-se apenas à uma transformação de estado de um sistema por outro, reduziria a definição a inutilidade de resumir um programa a qualquer outro conjunto de coisas no universo que interfere sobre outro conjunto de coisas.

Um programa é, mais especificadamente, uma tecnologia de automação, de transferência da execução de um algoritmo do domínio do pensamento para outro domínio, que, em geral, chamamos de físico.

Desta feita, o conceito de programa fica fundamentado em parte no domínio do pensamento, na forma de algoritmo, e no domínio físico, na forma de código, que, de forma análoga ao pensamento, existe no contexto de uma linguagem física. Essa divisão do conceito em dois domínios implica na existência de uma interface de comunicação.

A definição de programa não apenas implica na existência de uma interface, mas também determina o sentido da comunicação através dela.

A intenção, na forma da definição do problema e da solução, é comunicada ao sistema físico através da interface, e, após a execução do código, o estado final alcançado é comunicado de volta através dela.

O layout da interface é então um lugar privilegiado para o início do desenvolvimento de um programa. Nela estão presentes, de alguma forma, todas as entradas e saídas, que são as bases dos algoritmos que devem compor o programa, e cujas disposições relativas refletem, em certo grau, suas relações entre si nos algoritmos. Ao planejar o layout que melhor reflete essas relações, encontra-se também a melhor forma de agrupar, ordenar e relacionar as diversas partes do algoritmo. Até mesmo o desenvolvimento das teorias pode ser auxiliado pelo planejamento do layout.

De posse do layout e do algoritmo e sua teoria subjacente, sabendo agora que é possível a execução, trata-se de encontrar os sistemas físicos e linguagens que sejam compatíveis e selecionar o mais vantajoso. Um desenvolvimento em outra ordem, como o de primeiro escolher o sistema e linguagem física, desenvolver código com o desenvolvimento em paralelo do algoritmo apenas implícito nele e com conhecimento apenas parcial das teorias subjacentes é um método irracional de desenvolvimento.

A ordem mais lógica é começar pela definição do problema, e então, teoria, algoritmo, escolha de sistema e linguagem, desenvolvimento do código. A escolha de sistema, linguagem e desenvolvimento do código já envolvem o desenvolvimento da interface.

O lugar privilegiado não é então a interface em si, mas seu layout, já que, independentemente de como será realizado, a relação entre seus elementos reflete parte da estrutura tanto do algoritmo, do código, especialmente a parte do código referente a interface. A complexidade do layout da interface é provavelmente sempre inferior a complexidade do algoritmo e do código, mas serve como guia para a estrutura mais geral de ambos.

Em relação ao não desenvolvimento paralelo de código e algoritmo, o que ocorre é que ao desenvolver o algoritmo diretamente na escrita do código, as complexidades de ambos se misturam facilmente e fica mais difícil separar os erros de um e de outro. O código contém todo o algoritmo necessariamente, mas o algoritmo não possui nenhum atributo próprio da máquina. A partir do momento que se tem a teoria do algoritmo aceita, todo problema durante o restante do desenvolvimento fica atribuído as partes do código referentes ao sistema físico e sua linguagem.