diferença entre os dois conceitos pode ser mais bem entendida através da definição dada por Boehm [Boehm1981]:

"Verificação: Estamos construindo o produto da maneira certa?"

"Validação: Estamos construindo o produto certo?"

A principal desvantagem do modelo cascata é que boa parte do sistema não estará disponível até um ponto adiantado no cronograma do projeto e, geralmente, é difícil convencer o usuário de que é preciso paciência. Além disso, existe a dificuldade de acomodação das mudanças depois que o processo está em andamento. Portanto, esse modelo é mais apropriado quando os requisitos são bem entendidos. No entanto, há também algumas vantagens associadas ao modelo, pois ele oferece uma maneira de tornar o processo mais visível, fixando pontos específicos para a escrita de relatórios e, didaticamente, é uma maneira mais fácil de introduzir os principais conceitos de Engenharia de Software. Por esse motivo, as atividades do ciclo de vida do software serão detalhadas nos capítulos 4 e 5 com base neste modelo de ciclo de vida.

3.2 O MODELO DE DESENVOLVIMENTO EVOLUCIONÁRIO

O Modelo de Desenvolvimento Evolucionário (vide Figura 3.3) subdivide-se em dois: Programação Exploratória e Prototipagem Descartável.

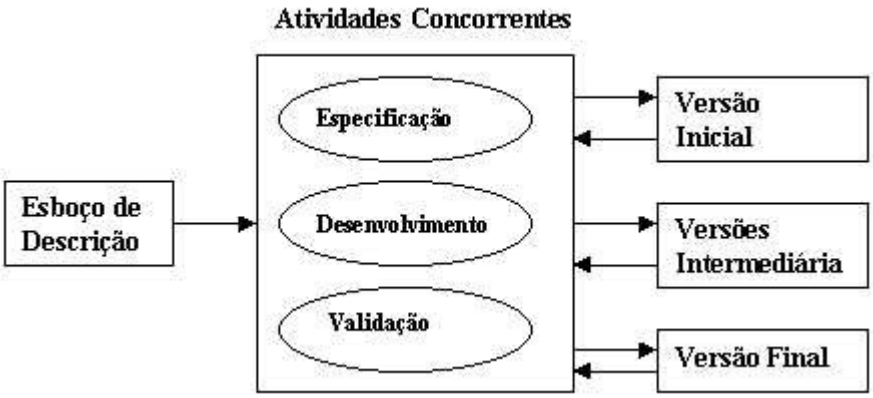

Figura 3.3: O Modelo de Desenvolvimento Evolucionário

3.2.1 O Modelo de Programação Exploratória

O objetivo desse modelo é o desenvolvimento da primeira versão do sistema o mais rápido possível. Os sistemas desenvolvidos com esse modelo caracterizam-se por não terem o escopo claramente definido, ou seja, a especificação do escopo é feita de forma intercalada ao desenvolvimento. Após o desenvolvimento de cada uma das

versões do sistema, ele é mostrado aos usuários para comentários. Modificações sucessivas são feitas no sistema até que o mesmo seja considerado adequado. A principal diferença dos outros modelos é a ausência da noção de programa correto. Esse modelo tem sido usado com sucesso para o desenvolvimento de Sistemas Especialistas, no contexto da Inteligência Artificial (ex: sistemas de reconhecimento de voz, sistemas de diagnóstico médico etc.).

3.2.2 O Modelo de Prototipagem Descartável

O objetivo principal desse modelo é entender os requisitos do sistema. Tem sido usado com sucesso para validar partes do sistema (Interface Gráfica e aspectos do sistema relacionados à arquitetura – ex: performance, portabilidade etc.). Como na programação exploratória, a primeira etapa prevê o desenvolvimento de um programa (protótipo) para o usuário experimentar. No entanto, ao contrário da programação exploratória, o protótipo é então descartado e o software deve ser re-implementado na etapa seguinte, usando qualquer modelo de ciclo de vida (ex: cascata).

3.3 O MODELO DE TRANSFORMAÇÃO FORMAL

Uma especificação formal (definição matemática, não ambígua) do software é desenvolvida e, posteriormente, transformada em um programa executável (vide Figura 3.4), através de regras que preservam a corretude da especificação (passos de refinamento).

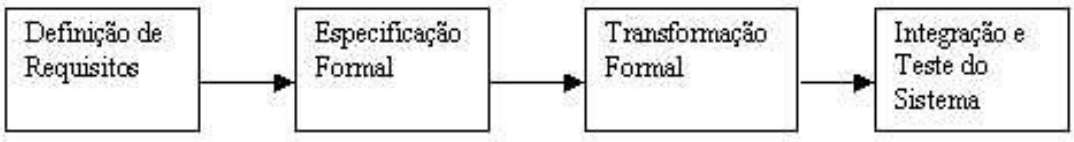

Figura 3.4: O Modelo de Transformação Formal

Esse modelo tem sido aplicado ao desenvolvimento de sistemas críticos, especialmente naqueles onde a segurança é um fator crítico (ex: sistema de controle de tráfego aéreo). No entanto, a necessidade de habilitações especializadas para aplicar as técnicas de transformação e a dificuldade de especificar formalmente alguns aspectos do sistema, tais como a interface com o usuário, são fatores que limitam seu uso.

3.4 O MODELO DE DESENVOLVIMENTO BASEADO EM REUSO

Esse modelo baseia-se no reuso sistemático de componentes existentes ou sistemas COTS (*Commercial-Off-The-Shelf*). Durante o projeto do sistema, os componentes que podem ser reusados são identificados e a arquitetura do sistema é