THE BOEING 777: NO CHAINSAW MASSACRES, PLEASE!

Niels Jørgensen
Department of Computer Science, Roskilde University, Roskilde, Denmark

Iterative life cycle models have become popular in software engineering, e.g. in agile development. In contrast, the waterfall model appears to prevail in manufacturing engineering disciplines. This includes aircraft engineering and Boeing's project for developing its most recent passenger aircraft, the 777. The paper walks through the phases of the 777's development and compares this process to iterative development. The comparison suggests two observations: Firstly, the over-all waterfall approach in the 777 project appears to be well-motivated by the physical, manufactured nature of aircraft components such as wings, in addition to safety concerns. Secondly, several iterative elements in the development of the 777 can also be identified. A major source of these is digitalization of development, in particular the use of CAD tools for a process called digital pre-assembly.

Keywords: *development life cycle, phased development, design change, aircraft engineering.*

1. Introduction

Maier and Rechtin in their book on systems architecting make several bold statements about life cycle models, including "hardware is best developed with as little iteration as possible, while software can (and often should) evolve through much iteration" (Maier and Rechtin, 2000, p 95). The present paper explores this and related assertions about life cycle models and their relevance for different domains of engineering. Life cycle models considered are waterfall and iterative models. Two highly simplified variants of these are shown in Figure 1.

It is not surprising that a large aircraft such as the Boeing 777 is developed using a waterfall life cycle model. The approach has prevailed in the American aerospace industry ever since the World War II era (see e.g. Ziemke and Spann, 1993). Phil Condit, project manager of the 777 project and later Boeing CEO, used the *bon mot* "no more chainsaws" to sum up the rationale of the waterfall model. This refers to the devastating effect of cutting up the fuselage of an almost completed aircraft to fit a changed part, and presumes that there is a way (i.e., the waterfall model) to design the part correctly in the first place.

In justifying a study of the usefulness of the waterfall approach for a project completed more than ten years ago, firstly it can be observed that the Boeing 777 is in fact the most recently developed large passenger aircraft in the world. Boeing's next model, the 787, is scheduled for delivery in 2008, Airbus' A380 in 2007.

Secondly, aside from the forseeable waterfall approach of the 777 project, the study identifies process elements of an iterative sort. These were related to digitalization of design, indicating a relationship between life cycle model and the software/hardware distinction, as asserted by Maier and Rechtin.

Thirdly, the pros and cons of the waterfall and iterative models remain disputed. The literature spans many views, including the following two opposing, extreme views: One is the *life-cycle-follows-*

artifacts view of Maier and Rechtin. The other is an *all-round-model view*. Several studies emphasize the all-round usefulness of a particular life cycle model. These include Auyang's account of the history of modern technology, which presents a phased, waterfall-type development model as the engineering method per se (Auyang, 2004). Symmetrically, Clark and Iansiti in their analysis of product development strategies in the 1990s, argue the universal usefulness of an iterative model, in the software industry as well as in manufacturing industries such as the automobile industry (Clark and Iansiti, 1997).

The paper is organized as follows: Sections 2-4 present the paper's method, provides background on life cycle models, and summarizes the business context of the 777 project. Section 5 walks through the 777 project's four development phases. Section 6 identifies and discusses waterfall-type elements, and Section 7 identifies and discusses iterative elements in the 777 project. Section 8 concludes.

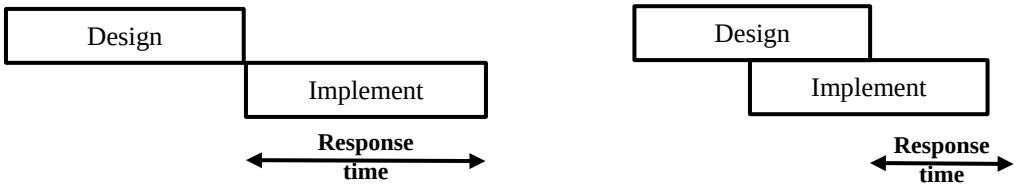

Fig. 1: Waterfall (left) vs. iteration (right). A benefit of the iterative approach is the ability to respond more quickly to design changes, and so adapt to changing markets and technologies.

2. Method

The method of the paper is a case study of the Boeing 777 development project from 1986 to 1995. The main source of empirical data is the detailed and extensive account of the project in (Sabbagh, 1995). Empirical data on the project is interpreted to indicate a typical waterfall approach: development proceeded in distinct phases and project organization was hierarchical in a manner that reflected the aircraft's hierarchical decomposition into wings, fuselage, etc. Data on the project's discourse indicates a thinking inspired by the waterfall-related discipline of concurrent engineering, e.g., the project's use of the phrase design-build teams for certain cross-functional teams.

Analytically, the focus is on how the 777 project approach handled design changes. This is of interest because a major purpose of the waterfall life cycle is to avoid (or limit the number of) design changes that are made late in the project. Section 5 looks at both 'early' and 'late' design changes, that is, changes made before vs. after the project's design freeze. To explore what it would have meant for the 777 project to proceed in a more iterative manner, the analysis also employs contra-factual reasoning of the following common-sense sort: Suppose (contrary to what happened) that there had been a test flight of an early prototype; then this would have posed a threat to the lives of the test pilots.

3. Background: Waterfall vs. Iterative Life Cycles

The difference between waterfall and iterative life cycle models is the ordering of development phases. In the two-phase setting of Figure 1, waterfall models would prescribe a single design-implement pass, while iterative models would prescribe a repetitive ordering with design-implement, design-implement, and so on. Many aspects of life cycle models escape this simplification, of course.

For example, the waterfall variant first proposed for software contained a feedback-loop to a previous phase (Royce, 1970). However, for clarity the paper identifies waterfall models with strictly phased models – corresponding, by the way, to the effect gravity has on the flow of water.

3.1. Waterfall Life Cycle Models

Waterfall models have been justified with reference to the tenet that *design changes are more costly the later they are decided*. In the software engineering discipline, waterfall models prevailed in the 1980s. A key argument in favor of waterfall models is the ability to avoid costs of fixing poorly designed code (Boehm, 1988). This is as opposed to the code-and-fix approach presumably followed in the early days of software. See Figure 2 below for an illustration of the costs in terms of time of late changes – in the worst case, a "chainsaw massacre".

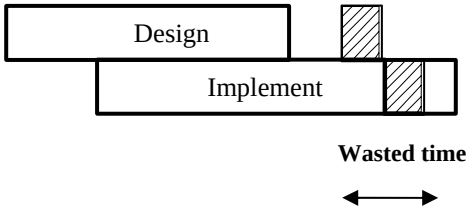

Fig. 2 A "chainsaw massacre" (indicated by hatched boxes) may result from the iterative approach, if the implementation is difficult to modify.

In manufacturing engineering disciplines, the literature of concurrent engineering provides an academic basis for the waterfall approach. Concurrent engineering can be defined as "simultaneous design of a product and all its related processes in a manufacturing system" (Jo *et al.*, 1993, p 4). Concurrent engineering emphasizes design-for-manufacture, design-for-assembly, and more generally, design-for-x, where x is integration, test, and other processes (i.e., phases). This is as opposed to design merely for end use. A major goal is to avoid redundant costs, such as when a product design poses manufacturing difficulties and thereby entails extra costs of manufacturing or re-design (Smith, 1997). – Note that in concurrent engineering, concurrency refers to designing with a view to multiple phases, and to simultaneous development of components (not to phase concurrency).

In aircraft engineering, key ideas of concurrent engineering were dissemminated by the Lean Aerospace Initiative (LAI). The process improvement effort at MIT was initiated in the early 1990s, at the time of the Boeing 777 project. For example, statistical data was publicized which showed that the design phase commits to two thirds of the full life cycle costs of a product. Costs committed by design include costs incurred during manufacture, maintenance, etc. This suggests a holistic, design-for-X approach to design (Murman *et al.*, 2000). – In addition to ideas related to concurrent engineering, the LAI incorporated ideas from lean automobile production. This is the implementation in Western countries of ideas suggested to explain the success of Toyota and other Japanese car makers (Womack *et al.*, 1991). This sense of lean includes a general focus on cost reduction, as opposed to cost reduction by design as in concurrent engineering. An example is just-in-time delivery of parts for assembly, one of many lean ideas which would appear to be independent of the ordering of development phases.

3.2. Iterative Life Cycle Models

Iterative models have been argued to enhance flexibility, debugging, and enthusiasm in a product development project.

Flexibility: In their analysis of the 'browser war' between Microsoft and Netscape in 1995-96, Clark and Iansiti stress the companies' ability to implement new features in six to eight weeks, allowing them to adapt to new user demands and new technologies (Clark and Iansiti, 1997). The short response time relied on releasing a series of prototypes, to the public (Netscape) or internally (Microsoft). In terms of Figure 1, response time is shortened because implementation provides an early prototype; the prototype generates design feedback, and its modular architecture facilitates the addition of new features.

Reduction of integration risk: McConnell in his book on rapid development argues that an iterative approach may support debugging during integration (McConnell, 1997). When parts are integrated into larger parts, errors emerge; their diagnosis is facilitated if the parts are added to a development version which is always kept in a working state. If adding a part entails a broken build or a failed regression test, the part is diagnosed to be involved in some interdependency problem. Implementation phases in rapid development are organized in small increments whose integration involves building and regressions testing on a frequent basis, say daily.

Enthusiasm: Brooks mentioned a psychological aspect of what he called growing of software: "Enthusiasm jumps when there is a running system" (Brooks, 1987, p18). Cusumano and Selby observed similar effects of incremental development at Microsoft (Cusumano and Selby, 1995). In an interview with this author, a software developer of the FreeBSD operating system said ".. there is a tremendous satisfaction to the 'see bug, fix bug, see bug fix get incorporated so that the fix helps others' cycle" (Jørgensen, 2005). Analogously to flexibility and debugging, a precondition is early implementation, providing a working prototype which is open for changes to be inserted and evaluated – and enjoyed.

3.3. Choice of Life Cycle as Depending Upon Artifact to be Developed

An engineer's knowledge about different life cycle models can be seen as a repertoire in the sense of Schön (Schön 1983). The engineering literature contains mappings of problems to life cycles, such as McConnell's list of when various software engineering life cycle models apply (McConnell, S., 1996). The choice of life cycle model is difficult because, among other reasons, there is a multitude of possibly relevant parameters:

- The independent variable side is the artifact to be developed, represented perhaps by an initial specification; here parameters include such factors as (estimations of) the complexity of the artifact and the required design effort. Perhaps this side includes also a project context in terms of a pool of participants, their culture, etc.

- The dependent variable side is the life cycle model; parameters include the specifics of the model, such as the selection and ordering of phases, type of phase transitions such as formal reviews, etc. See Figure 3 below.

These variables are difficult to define conceptually, let alone quantify and measure. Empirical data is from case studies that resist generalization. Difficulties increase further when making assertions about artifact / life cycle dependency across different engineering domains.

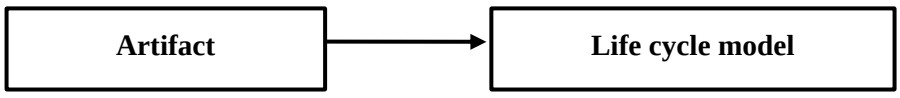

Fig. 3 Artifact to be developed as independent variable and life cycle model as dependent variable.

In Maier and Rechtin's discussion of the difference between software-intensive systems and manufactured hardware systems, three assertions can be identified (Maier and Rechtin, 2000, Chapter Six), which detail the assertion quoted in the introduction.

1. Hardware is best developed with as little iteration as possible. This assertion is discussed in the present study.

2. An iterative approach is possible with software. Complementing the first assertion, this is due to the low cost of software copying and distribution. Maier and Rechtin focus on field upgrades, but the argument applies to the development organization as well. For example, iterative development at Netscape, Microsoft, and FreeBSD rely on two modes of copying and (internal) distribution: (1) the full system to the local site (for trial integration of a new change), and (2) subsequently, the change to the central unit (for final integration), as discussed in (Holck and Jørgensen, 2004).

3. Software can be designed with a modular architecture, which facilitates iterative development. This is as opposed to hierarchical product architecture, where a system is composed of subsystems and so on recursively. While software can also be designed hierarchically, its flexibility allows for alternatives. Examples of non-hierarchical software architectures given by Maier and Rechtin include layered and object-oriented structures.

Assertions made in the software engineering literature include the following

4. An iterative approach is suitable (only) if there is already an established design. The availability of a sound design was suggested by McConnell as an explanation of the success of the Linux project's iterative approach which had no dedicated design phase. "By the time Linux came around [..] architecture defects had already been flushed out during the development of many previous generations of Unix" (McConnell, 1999). This argument may not apply to iterative approaches per se, and in particular not to Boehm's iterative and structured model, the spiral model (Boehm, 1988).

5. An iterative approach is suitable (only) if the project is of limited size. Kenn Beck, coauthor of the agile manifesto (Beck, 2001), argues that extreme programming (XP) with its iterative approach is appropriate only for projects of less than about 20 developers. Beck asserts that the integration of all changes into a single development version is XP's scaling bottleneck (Beck, 2000). A waterfall approach provides the alternative of staged integration, i.e., unit integration, component integration, etc. This limits the number of parts which are assembled with each other in each stage. The present author's study of FreeBSD indicated that in phases of intense development, the project's development version was indeed overloaded with contributions. This led to failed builds, so that at times there would be no updated, working prototype. As a remedy, the project introduced design freeze periods, where only bugfixes were allowed. Rejecting new design contributions would help fix the basic prototype, but eroded the advantages of debugging and enthusiasm (Jørgensen, 2005). For a summary of the life cycle models and the five assertions, see Table 1 below.

It is tempting to try to identify a single, fundamental issue from which the differences between the life cycle models originate. I would suggest that the fundamental issue is how technological uncertainty is resolved. That is, whether one uses analysis or experiment to predict the performance of a proposed design, when performance cannot be inferred directly from existing technological knowledge. Evaluating by *experiment* yields the iterative approach, where direct feedback is obtained from a working prototype. Evaluating by *analysis* yields the phased approach, where a proposed design is studied by predictive modeling – in the design phase, prior to implementation. For example, the aerodynamic properties of a rudder design can be analyzed by computer modeling and wind tunnel testing. Vincenti's concept of vicarious modeling (Vincenti 1990) captures the fact that such analysis is a substitute for testing an implementation of the design. The rationale of vicarious modeling is to avoid costly, time consuming experiments with a full-scale implementation, and this is certainly a rationale of the waterfall model as well.

Table 1 Summary of waterfall vs. iteration.

	Waterfall	*Iterative*
Phase ordering	One pass	Multiple passes
Advantages	Handles complex design problems; design-for-X reduces cost	Flexibility; debugging; enthusiasm
Typical application	Manufactured technologies	Software-centered technologies
Limited application	Markets with rapidly changing requirements	Large projects; new, complex design problems

4. Background (continued): The Business Context of the Boeing 777 Development Project

The first Boeing 777 was delivered to United Airlines in May 1995 and entered commercial service the following month. Work on the 777 project commenced in 1986 as a builder-initiated project, to use a term from (Maier and Rechtin, 2000). The nine year time span from initiation to first delivery is an indication of the long-term nature of the huge investments made by Boeing, and the pressure to shorten and optimize the development process.

The period in which the 777 was developed witnessed fierce competition in the aircraft industry. Military budgets were cut upon the end of the Cold War around 1989, which resulted in a reduced demand for military aircraft. Privatization of airlines since the 1980s made them less loyal to local, government-supported aircraft suppliers. Competition also intensified when it became customary that aircraft were offered with a choice of engine supplier. Then an airline could choose freely among aircraft suppliers and still retain a single engine supplier with a familiar maintenance program.

At project initiation, mergers had left only two other suppliers of big commercial airplanes, McDonnell and Airbus. (Subsequently, McDonnell merged with Boeing in 1997.) Both of Boeing's competitors were developing new aircraft for the same market segment as the 777.

The end result for Boeing was an imperative to compete on cost and performance. The aircraft market may not be as dynamic as the web browser market in terms of new user requirements. However, freezing the design several years before delivery increases the company's response time viz-a-viz new developments in technology and market, such as rising fuel costs. If the waterfall approach is *necessary* in the highly competitive aerospace industry, it is a necessary *evil*.

5. Design Changes in the Four Phases of the B777 Project

The two main activities in the 777 project can be said to be design and build, corresponding to the concepts design and implement of Figures 1 and 2 above. The concepts of design and build were central in the project's own terminology, most notably in the name Design-Build Teams. Design-Build Teams (DBTs) was the key forum for making design decisions. There was a DBT for every component of some complexity, totaling approximately 250 teams. For example there was a passenger door DBT and a cargo door DBT.

The Design-Build Teams were cross-departmental, comprising design engineers, manufacturing engineers, maintenance engineers, and others from Boeing, as well as representatives from subcontractors and customer airlines. The idea was understood as one of avoiding departmentalization, where designers would throw designs over the wall to manufacturers. Instead, designers should involve

manufacturing people and use their knowledge. This is a straightforward application of design-for-x principles of concurrent engineering.

More specifically, four development phases of the 777 can be identified: Conceptual Design and Component Design as subphases of design, and Assembly and Test as subphases of build (see Figure 4). The four phases are identified for the purpose of the presentation, and do not reflect an explicitly stated Boeing life cycle. Subsections 5.1-5.4 discusses four design changes, one from each phase.

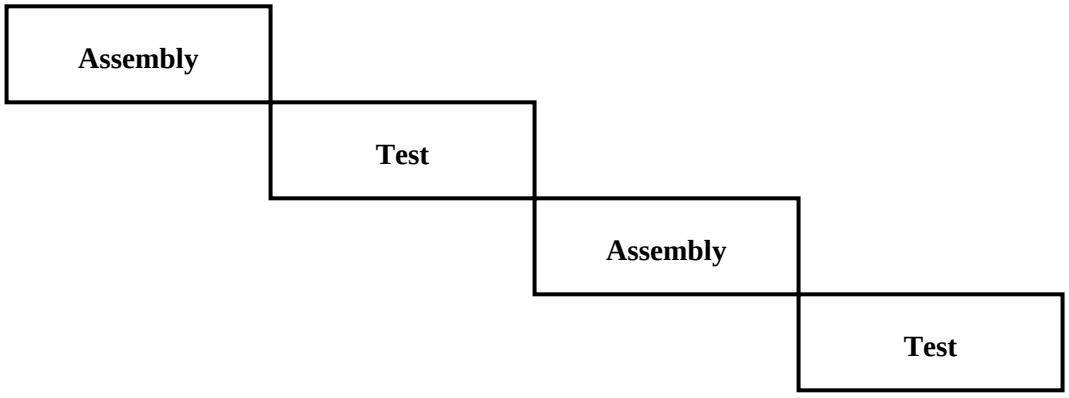

Fig. 4 The four phases of the 777 project.

5.1. Phase 1. Conceptual design, 1986 – October 90: From Derivative to Independent Model

Work on the 777 began with Boeing's investigation of how the company could enter the emerging segment of aircraft between the company's 747 (Jumbo jet) and its second-largest model, the 767. Aircraft in this segment would carry 300-400 passengers on long distance flight.

The main change decided on in the phase of conceptual design was to develop a new major model rather than an enlarged version of the 767. In 1988 the company decided that requirements had accumulated to above of what could be met by a 767 variant. The development of a new major model is a much greater undertaking than the evolutionary approach, and accordingly there was a re-definition of the project and its deadlines.

Variants that have been launched by Boeing include several versions of the 747 (Jumbo jet). The 747-400 entered service in 1989, nineteen years after the original 747, and has an improved wing design. Major new models are developed less frequently than variants. The 777 was the only major new model completed in the 1990s; the most recent major model, the 767, had entered service in 1982. In a sense, model variants represent incremental or evolutionary development, with the qualification that increments can be quite substantial.

The over-all configuration of the 777 comprised two engines and two aisles, similarly to the 767. Among the most significant new features of the 777 was fly-by-wire. The 777 was to be Boeing's first plane where pilots would control the rudder and wing flaps electronically. Additionally, the conceptual design comprised the size, shape, and materials used for major components such as the wings. Conceptual design can be viewed as effectively frozen in October 1990, when United Airlines ordered 34 planes at a price of approximately 100 million USD each. The contract also stated requirements about load, cabin and seat space, speed, and fuel consumption.

5.2. Phase 2. Component Design, October 1990 - January 1993: The Door Hinge Design Changes

The doors were designed in-house at Boeing. Almost all components of the aircraft were designed by Boeing, the major exceptions being the engines and the main IT-system.

The door design was an example of the 777 project's focus on reducing the number of design changes, as compared to the previous model, the 767. More than 13,000 design changes had been made to the design of the predecessor's doors. The company estimated the cost of these to 64 million USD, which is about half the sales price of a full plane. The main strategy was to use a common base of parts, so that there would be fewer parts to change. Eventually the passenger doors would use 98% common parts.

Using a common hinge for all eight passenger doors was a challenge because the shape of the doors is not the same. The doors sit in the fuselage at places where the diameter is different. During the component design phase, the fuselage shape was altered twice (for reasons related to the plane's overall aerodynamic performance). This implied a need to change the shape of the doors, and in turn, their common hinge.

The complex nature of the hinge design task is indicated by the set of partly conflicting requirements that had to be taken into account. There are generic constraints pertaining to weight and outer surface form and smoothness. Specific requirements include a closing mechanism to prevent opening during normal flight, sealing to prevent loss of cabin air pressure, and strength of the door as a whole to withstand the force exerted by cabin air pressure. The force amounts to 15 metric tons, given the area of the door and including a safety margin. At the same time, the door hinge must be easy to turn, even by not so strong hands with long fingernails.

From the point of view of the waterfall philosophy, the door hinge design changes were benign, because they occurred before completion of the design phases. Each time the fuselage shape was altered, a new design of the doors and the common hinge was completed. In the first round, the hinge was designed in three months. In the process, the engineers almost gave up on the goal of using a common hinge. However, the second round took only one month and the third round only a week. The chain of events was reflected on by the engineers involved as a process of build-up of competence to react to changing requirements. The door hinge design challenges were solved in a small-scale iterative process within the overall phased approach of the project.

5.3. Phase 3. Assembly, January 1993 - March 1994: The Rudder Design Change

Wing assembly started at Boeing's factory in Everett, Seattle, in January 1993. Other parts would be assembled later or earlier, depending on their position in the product hierarchy (see Figure 5). For example, the major components of the wing are wing box, leading edge, and trailing edge. The wing box contains spars, which are 30 meter long and had been assembled themselves before they were built into the wing box, etc. Thus the suggested assembly start date is somewhat arbitrary.

The rudder design changes are interesting because they were made in January 93, which was several months after the promised design freeze date, and because the rudder, while designed in-house by Boeing, was assembled by an Australian subcontractor (ASTA).

The main challenge in the design of the rudder was strength. Each engine of the Boeing 777 is extremely powerful, and if one fails, say the left engine, then the functioning engine which is attached to the right wing will turn the aircraft to the left. To balance the uneven engine thrust, the movable part of the rudder must be swung to the right side. The actuator that moves the rudder must apply a strong force, and the rudder and its hinges must transfer a strong force back to the plane once the air stream starts exerting pressure on it. The final design change introduced a small bulge on the surface of the rudder, a compromise that made room for a strengthened actuator at the cost of increased drag (and fuel consumption).

The rudder was built using almost a thousand purpose-made tools. The design change was so comprehensive that the majority of the tools had to be rebuilt as a consequence. Eventually the subcontractor solved the re-tooling challenge and managed to deliver the rudder at the date originally agreed on.

There are obvious differences between assembly in manufacturing engineering disciplines and assembly of software. In manufacturing, assembly is intertwined with what may be called unit process fabrication (Whitney, 2004). This is processing of individual parts, as opposed to assembly with other parts. For example, the rudder was baked in a large oven, a process lasting for seven hours, to harden the carbon fibre surface. Advanced machine tools were widely used in the assembly phase, and to the extent that a tool for a part depends on the detailed design of the part, assembly must await completion of the tool. In contrast, tools used in software development, such as IDEs (Integrated Development Environments) are more universal, and less dependent upon design changes of the product.

5.4. Phase 4. Test, March 1994 - June 1995: Engine Backfire

The first plane was weighted on 18th March, 1994. This can be seen as the first major test of the plane. The assembled plane's weight was a key performance parameter. The test showed 135 metric tonnes against a predicted value of 132. The first test flight was June 12, 1994, followed by a full year of flight testing.

To speed up engine testing, an engine from Pratt & Whitney of the type to be fitted on the firstly delivered 777, was fitted to an old 747. This was in November 93 when the first 777 was only in assembly.

The flight-test of the engine revealed a serious flaw. The engine showed behavior reminiscent of "backfiring" (a so-called engine surge), causing the aircraft to stall temporarily. Flight testing on the 747 provided more time for Pratt & Whitney's engine re-design than if flight testing had awaited the readiness of the 777 for flight. Test-flying the 747 with the new engine is reminiscent of an evolutionary or incremental approach, since it provides design feedback from a full scale prototype. This was possible with the engine because unlike components such as rudder and wing, it was feasible to retro-fit the engine to another aircraft – although retro-fitting was costly, and was undertaken only when the chief test pilot insisted on early engine testing in real flight.

Table 2 below summarizes the four design changes discussed above. The first two changes occurred during the design phases, and so were consistent with the strategy, while the two others were of the post festum type.

Table 2. Summary of four design changes in phases 1-4.

Phase	***Component***	***Approach to design change***
1. Conceptual design	(Entire aircraft)	(Launch of major new model)
2. Component design	Door hinge	Competencies built in three design rounds
3. Assembly	Rudder	Cost of tool re-design deferred to sub-contractor
4. Test	Engine	Early detection by prototyping with old plane

6. The Waterfall 777 Approach and the Physical Nature of Aircraft Components

Components of an aircraft such as wing and rudder are artifacts of a physical nature. This section discusses how this nature lends itself towards a phased development process. Each component exemplar represents a value, reflecting its manufacturing cost. There is also a cost associated with transporting a component from the place of its assembly to the place where it is assembled into a larger component. In contrast, the cost of copying and distributing a software artifact is negligible.

The significance of the cost of individual exemplars of a component is illustrated by the 777 wing testing. Wing testing culminated in January 1995 with the so-called snap test, a year into the test phase. The wing is required to withstand a load corresponding to the plane's weight plus an additional 150%. In a controlled, indoor environment, an increasing force is applied to lift the wing tips higher and higher, while the fuselage remains in a fixed position. Eventually something breaks. In the snap test, the wing broke at 157%, yielding a successful test.

The wing snap test, being a destructive test, required a plane taken out of the production line of the first series of planes. Both wings of the test plane broke and were rendered useless, except for providing data about the breakage. (Even if the snap test had been aborted just above the 150% level, the wings would already have been deformed.)

The cost of a broken aircraft wing greatly exceeds those of a broken build of a software project. The information provided by a build breakage (namely that a module added since the previous successful build has introduced an error) incurs a loss of development time only. Broken software builds during development do not destroy physical products, let alone costly products such as wings of a 100 million USD aircraft.

The significance of the cost of component transportation is indicated by the first rudder delivery in August 1994 to the main assembly site in Seattle. The first rudder was flown in using a 747 freighter (into which it would just fit), to save five weeks of seaway transportation (as used subsequently). Even though there remained a full seven months to assembly completion and 11 months to first flight (see Phase 4 above), presence of the rudder at the assembly site was essential for the time plan. This indicates the relevance of the "one-shot" approach. An iterative approach to fitting the rudder onto the plane's tail is prohibited in these circumstances, if such an approach requires awaiting the arrival of modified rudders.

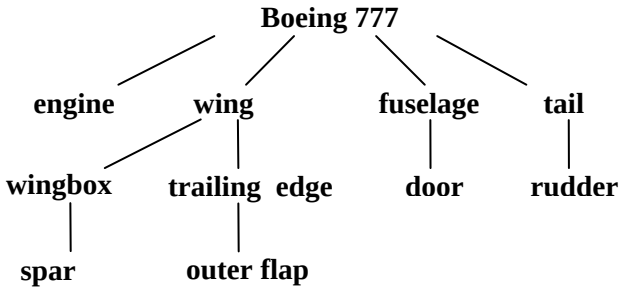

Fig. 5 Hierarchical product view of the B777. There was a separate Design-Build Team (DBT) for spar, outer flap, door, and rudder. The DBTs reported upwards in a project organization mirroring the product hierarchy. Design was sent down to local sites for production. Parts were transported reversely for assembly.

Geographical distribution of component manufacturing was the rule in the 777 project. The components discussed in this paper, shown above in Figure 5, were produced at distant sites. The doors were produced in Japan (by Mitsubishi) and one engine alternative in England (by Rolls Royce). Spars were produced by a subcontractor located 30 miles away from the wing assembly plant, but still, transportation of the 30 meter long spar was somewhat cumbersome. It may be noted that Boeing's coming 787 model will have its entire wing assembled in Japan, and already the Airbus company transports fully assembled wings from England to the final assembly plant in France.

7. Iteration in the 777 Project and CAD-Based Development

Boeing referred to the 777 as the first fully digitally designed plane. The present section first describes the role of the CAD tool CATIA in Boeing's design process, and then discusses how this introduced iterative elements into the 777 development process.

7.1. Digitalized Design

The 777 design was defined fully in digital form, and communicated digitally among participants, inside and outside of Boeing, rather than as paper drawings. The CATIA (Computer-graphics Aided Three-dimensional Interactive Application) CAD tool from Dassault and IBM included the following features:

Animation could be done in three dimensions, for example showing the cockpit or a maintenance area from arbitrary points of view. The movement of some parts could also be simulated, for example opening of doors. Additionally, the CAD tool had analytical capability to predict in an approximate manner various properties of a component, such as weight and strength of a piece of metal cut out in a certain way. A significant part of the CATIA tool's analytic capabilities was the so-called digital pre-assembly feature. Pre-assembly is prediction in advance of actual assembly of whether parts will fit together.

At one instance in March 1992, towards the end of the design phase, the pre-assembly feature was run on the most current design of the outer flap. The outer flap is a component of the wing's trailing edge (see the component hierarchy in Figure 5 above). The purpose of the flap is to provide extra lift. This is useful at the lower speeds desirable for take off and landing. The flap is movable, and when moved out extends the wing to the rear and somewhat downwards; as higher speeds, the flap is rolled back into the wing's trailing edge.

When asked to check twenty major components of the outer flap, the system found 251 interferences between the components. A typical interference is a physical overlap, where two parts occupy the same physical space. There are also subtle inter-dependencies such as when a part extends into another part's so-called swept volume. This is space surrounding a part that must be kept empty, for example to allow the part to be taken out for replacement without removing other parts to gain access.

Detecting interference among parts in the outer flap by digital pre-assembly during the design phase allowed parts to be re-designed well ahead of actual assembly. Indeed digital pre-assembly was seen as crucial to get the design right before assembly, a key goal of Boeing's phased approach.

7.2. Analysis of Digital Pre-Assembly

Pre-assembly is useful because of the extremely high number of parts. The designer of a part can not capture its possible inter-dependencies with all other parts.

Digital pre-assembly of the 777 replaced a method of pre-assembly by physical mock-ups, which had been used previously, e.g., on the 747 and the 767. The mock-ups were full scale, non-flying versions of the planes in easy to build materials, such as foam and plywood. Typically, three stages of mock-ups would be used, in increasing detail and using a larger share of real parts. Digital pre-

assembly allows for checking parts interference with greater accuracy. Also the digital processing allows for more frequent checking, since the time consuming physical construction of mock-ups is eliminated.

Digital pre-assembly as practiced by the 777 development project in several ways resemble iterative software development.

Firstly, all design data was stored centrally, resembling a software project's central repository containing the project's development version in source form.

Secondly, the central storage was easily accessible to designers, for inspection of the design of other parts. However CATIA ran on a cluster of central computers, whose computing power was as scarce resource, so an individual designer or team could not invoke frequent interference checking of proposed designs. Rather, interference checking of a component was done on a single variant of the design, the centrally stored variant.

Thirdly, the project used digital pre-assembly in an iterative manner, comprising six rounds of design separated by six design freezes. Design freezes meant that designers were prohibited from entering new designs into the central store of the CATIA tool, a measure that spurred complaints from designers. The only access allowed during design freeze was to insert changes to sort out parts interference. The repetitive design freezes in the 777 project resemble design freezes as used in iterative software development. In the latter, design freezes address the problem of overloading the development version of the software – it cannot be used for stabilization and new development simultaneously. Indeed, similarly as in the 777 project, developers in Mozilla and FreeBSD have complained that design freezes were blocking their design work (Holck and Jørgensen, 2004).

Vincenti's notion of vicarious modeling (Vincenti, 1990) may be useful in characterizing digital pre-assembly (see also Section 3 above). A key element of technological progress, according to Vincenti, is the evolution of vicarious modeling to attain greater predictive power, as exemplified by the achievements of Computational Fluid Dynamics, which has increased the usefulness of computer models of a wing's drag and lift. Vicarious modeling in Vincenti's sense applies to predictive evaluation of the external performance of a part or entire product, such as lift of a wing or drag of a full plane. However, the concept may be extended to the internal characteristics of an assembly, such as whether the parts of an assembly will overlap. Thus, the replacement of mock-ups with digital pre-assembly is an evolution from a physical to a digital form of vicarious modeling of the assembly process. In geographically distributed development, the problem of assembling parts originating from diverse suppliers is of increasing importance, and so are efforts to improve the assembly process by modeling.

8. Conclusion

A walk through of the Boeing 777 project of 1986-95 indicates that the project followed a waterfall life cycle. Long phases of conceptual and component design (each 2-3 years) were followed by shorter phases of assembly and test (2.5 years in total). The project employed Design-Build Teams and a discourse of cross-departmental cooperation, consistently with concurrent engineering.

The phased 777 approach seems well suited to the nature of the aircraft components involved. The cost of their manufacture and worldwide transportation indicates the importance of getting design right before assembly. In addition to this, safety concerns prohibit test flying of immature prototypes, of course. The introduction of new software technology, most notably fly-by-wire, does not appear to outweigh the physical nature of the basic flight components. The plane appears to fit with Maier and Rechtin's rule of thumb that hardware-centered technologies contains a 70% hardware part against a 30% software part (Maier and Rechtin, 2000, p 89).

Despite the overall phased approach, there were also identify iterative elements of the B777 development process. The changing door hinge requirements spurred three rounds of design and re-

design which were reminiscent of iterative development. The completed engine had a design error that led to "backfiring", and this was identified relatively early by testing the engine on an old Boeing 747 plane, which can be seen as a form of prototyping. Most notably, digitalization of design introduced consecutive design freezes for stabilization of intermediate designs, which resembles design freezes in iterative software development. Digital pre-assembly can be interpreted as vicarious modeling of the assembly process. Interestingly, since the rationale of vicarious modeling is to predict during design, digital pre-assembly is consistent with the world-view of concurrent engineering, so digital pre-assembly is related to the waterfall as well as the iterative life cycle model.

References

Auyang, S.Y., 2004, "Engineering - an Endless Frontier. Harvard University Press," Cambridge, MA.

Beck, K., 2000, "Extreme Programming Explained," Addison-Wesley, Reading, Massachusetts.

Beck, K., et al., 2001, "Manifesto for Agile Software Development," URL: http://www.agilemanifesto.org/.

Boehm, B.W., 1988, "A Spiral Model of Software Development and Enhancement," Computer 21 (5), May: 61-72.

Brooks, F.P., 1987, "No Silver Bullet. Essence and accidents of software engineering," IEEE Computer, April: 10-19.

McConnell, S., 1996, "Rapid Development," Microsoft Press, Redmond, Washington.

McConnell, S., 1999, "Open-Source Methodology: Ready for Prime Time ?" IEEE Software, 16 (4), 6-11.

Cusumano, M.A., and Selby, R., 1995, "Microsoft Secrets," Free Press, New York.

Holck, J. and Jørgensen, N. 2004, "Do Not Check in on Red: Control Meets Anarchy in Two Open Source Projects," in: Stefan Koch (ed), Free/Open Source Software Development. Idea Group Publishing, Hershey, PA, USA: 1-26.

Iansiti, M. and MacCormack, A., 1997, "Developing Products on Internet Time," Harvard Business Review, 75 (Sep-Oct): 108-117.

Jo, H.H., Parsaei, H.R., and Sullivan, W.G., 1993, "Principles of Concurrent Engineering," in: Parsaei and Sullivan (eds), Concurrent Engineering. Chapman & Hall, London.

Jørgensen, N., 2005, "Incremental and decentralized integration in FreeBSD," in: Feller, J., et al. (eds), Perspectives on Free and Open Source Software, MIT Press, Cambridge, Massachusetts, USA: 227-243.

Maier, M.E. and Rechtin, E., 2000, "The Art of Systems Architecting," (2nd ed.), CRC Press, Boca Raton, FL.

Murman, E.M., Walton, M., and Rebentisch, E., 2000, "Challenges in the Better, Faster, Cheaper Era of Aeronautical Design, Engineering and Manufacturing," Aeronautical Journal, 104 (October): 481-489.

Royce, W. W., 1970, "Managing the Development of Large Software Systems: Concepts and Techniques," Proceedings of IEEE WESCON, August: 1-9.

Sabbagh, K., 1995, "21st Century Jet. The Making of the Boeing 777," Macmillan, London.

Schön, D., 1983, "The Reflective Practitioner," Basic Books, New York.

Smith, R.P., 1997, "The Historical Roots of Concurring Engineering Fundamentals," IEEE Transactions on Engineering Management, 44 (1), February: 2-13.

Vincenti, W.G., 1990, "What Engineers Know and How They Know it. Analytical Studies from Aeronautical History," Johns Hopkins University Press, Baltimore, MD.

Whitney, D.E., 2004, "Mechanical Assemblies," Oxford University Press, New York.

Womack, J.P., Jones, D.T., and Ross, D., 1991, "The Machine That Changed the World: The Story of Lean Production," Harper, New York.

Ziemke, M.C., and Spann, M.S., 1993, "Concurrent Engineering's Roots in the World War II Era," in: Parsaei and Sullivan (eds), Concurrent Engineering. Chapman & Hall, London.