# 1. Einleitung

## 1.1 Das Kochbuch

Mit dem vorliegenden Buch wollen wir eine Art "Kochbuch" für Manager, Projektleiter, Anwendungsentwickler und Benutzer schaffen, denn mit einem Kochbuch kann man dank der einfachen Anleitung ein gutes, vielleicht sogar ein nicht alltägliches Gericht kochen. Aus der Abbildung ersehen Sie, dass wir unser Vorgehensmodell dem allgemein gültigen Aufbau eines Kochbuches angepasst haben. Im Kapitel Grundlagen finden Sie eine Menge von Begriffen und Definitionen, die Sie in jedem Entwicklungsprozess anwenden können. Die einzelnen Prozesse sind immer als Übersicht und im Detail beschrieben. Weiter finden Sie eine Fülle von Praxisanwendungen sowie eine Fallstudie als Anleitung. Mit diesen Grundlagen sollten Sie in der Lage sein, ein vorbereitetes "Gericht" zu kochen oder ein neues selbst zusammenzustellen.

## 1.2 Geschichte

Das traditionelle ("Wasserfall")-Vorgehensmodell, bei dem eine Phase sequentiell einer anderen folgt, hat sich leider als wenig tauglich herausgestellt:
Der Evolutionszyklus geht davon aus, dass die Benutzer ihre Phase(n) vollständig definieren können. Anhand von Diagrammen und verbalen Beschreibungen muss der Auftraggeber (Abteilung, Benutzer) entscheiden, ob er mit den Spezifikationen einverstanden ist. Bekommt er dann das fertige Produkt, nach allzu langer Zeit zu sehen, entspricht das Ergebnis sehr oft nicht seinen Vorstellungen. Was ist die Konsequenz? Die Wartungsphase dient während der Einführung vor allem dazu, geänderte und falsch verstandene Anforderungen nachzuvollziehen und zu implementieren. Bis die Applikation dann effektiv in Produktiion gehen kann und auch genutzt wird, vergeht enorm viel Zeit. Die Benutzer sind unzufrieden, da die von ihnen benötigten Programme nicht nur zu spät ausgeliefert werden, sondern auch nicht der gewünschten Funktionalität entsprechen. Markt- und produktbezogene Ergänzungen sind nur noch mit grossem Aufwand realisierbar und verzögern die Markteinführung neuer Produkte und Dienstleistungen, was einer verminderten Wettbewerbsfähigkeit gleichkommt.

Um die Applikationsentwicklung effektiv zum Instrumentarium der Wettbewerbsfähigkeit zu machen, ist ein anderes Vorgehen nötig, damit die Hauptforderungen erfüllt werden können:

- Frühzeitige Einbeziehung der Endbenutzer
- Kurzer Entwicklungszyklus
- Änderungs- und erweiterungsfreundliche Applikationen

Die Applikationserstellung in einem iterativen und evolutionären Vorgehen erfüllt diese Forderungen. Unternehmen werden so in die Lage versetzt, den Applikationsrückstau abzubauen und die Änderungen zur richtigen Zeit ("Just in time") zu berücksichtigen:

### Kurzer Entwicklungszyklus

Zunächst wird ein Kernteil der Applikation erstellt (Vorhaben mit Zyklen und Subzyklen) und unter Produktionsbedingungen praktisch eingesetzt. Diese Grundlage dient der Feinabstimmung zwischen Entwickler und Benutzer. So kann eine in Entwicklung befindliche Anwendung in die Produktion übernommen werden, sobald sie erste Nutzeffekte verspricht und nicht erst nach Abschluss der gesamten Entwicklung. Aufbauend auf diesem Kern wird weiterentwickelt und kontinuierlich an die weiteren Benutzeranforderungen angepasst. Die Phasen werden in kurzen Folgen durchlaufen. Das Resultat des Vorläuferzyklus ist die Basis für den Nachfolger.

### Einbeziehung der Endbenutzer

Bei der evolutionären Anwendungsentwicklung ist der Entbenutzer gleichzeitig Mitglied des Entwicklungsteams. Er arbeitet eng mit dem Entwickler zusammen und bestimmt so "seine" Anwendung mit. Damit wird sichergestellt, dass die Benutzeranforderungen voll umgesetzt und laufend den neuesten Anforderungen angepasst werden. Der Benutzer kommt schneller und kostengünstiger zu der von ihm gewünschten Funktionalität. Er kann sich mit der ausgearbeiteten Lösung identifizieren und es entstehen keine Akzeptanzprobleme. Für die traditionelle Vorgehensweise spricht man in der Industrie von einer seriellen (Fliessband-) Fertigung. Das evolutionäre Prototyping mit den ihm inheritären Gruppenanforderungen entspricht bei diesem Vergleich der Fertigung in autonomen Arbeitsgruppen.

### Änderungs- und erweiterungsfreundliche Applikationen

Objektorientierte, iterativ und evolutionär gestaltete Systeme können sehr leicht an sich ändernde Bedingungen (z.B. Benutzeranforderungen, neue zu unterstützende Produkte, Gesetzesänderungen) angepasst werden. Wartung und Erweiterung werden nicht mehr zur lästigen Pflicht, sondern gehören integrierend zum Lebenszyklus. Zudem entfallen weitgehend anforderungsbezogene Wartungsarbeiten bei der Einführung der Applikation.

### Vorteile der evolutionären Entwicklung

- Investitionsschutz durch kontinuierlichen Aufbau
- Keine grossen Schwankungen

# 1.3 Das Vorgehensmodell OOW

Warum ist unser Vorgehensmodell objektorientiert? Der Nutzen der Objektorientierung mit ihren Grundkonzepten Kapselung, Vererbung und Nachrichtenverarbeitung wird heute allgemein anerkannt. Auch die Möglichkeit, durch Definitionen (Klassen) und Nutzung immer wieder vorkommender Abläufe und Zustände (Muster) die Wiederverwendbarkeit zu steigern, wird genutzt. Warum soll also nicht auch ein Vorgehensmodell auf diesen Konzepten basieren? Unser Vorgehensmodell "Der Objekt-Orientierte Weg" OOW basiert auf diesen Konzepten. Es ermöglicht damit erleichterte Problemlösungen im organisatorischen, technischen und betriebswirtschaftlichen Umfeld. Der OOW selbst löst die Probleme nicht, zeigt dem Anwender aber auf, wie und in welchen Schritten er diese lösen kann.

## 1.3.1 Die Grundlage

Objektorientierte Technologien entstanden mit der Zielsetzung, eine hohe Wiederverwendbarkeit zu erreichen und diese aktiv zu unterstützen. Dazu sind folgende Mechanismen unabdingbar:

- Zerlegbarkeit
- Kombinierbarkeit
- Verständlichkeit
- Beständigkeit

Wenn wir diese Prinzipien auf das Vorgehensmodell übertragen, und ihre Forderungen erfüllen, dann können wir von dieser Philosophie auch dort profitieren. Da wir uns in der realen Welt immer nur mit Klassen und Objekten beschäftigen, muss dies auch in der Verfahrenstechnik möglich sein.

**"Der Objekt-Orientierte Weg" ist die Lösung dazu.**

## 1.3.2 Der Nutzen

**Hohe Stabilität**

Wenn Software aus existierenden, bewährten und geprüften Komponenten zusammengesetzt werden kann, führt dies zu qualitativ hochwertigen Systemen, die weniger Fehler aufweisen und besser auf die Geschäftsbedürfnisse abgestimmt sind. Da individuelle Objekte geändert und ergänzt werden können, ohne andere Objekte zu beeinflussen, können objektorientierte Applikationen leichter modifiziert und gewartet werden, als dies bei konventionellen Applikationen der Fall ist. Dies reduziert einerseits die Wartungskosten und erlaubt es andererseits, Softwaresysteme zu bauen, die sich besser den sich verändernden Geschäfts- und Umweltbedingungen anpassen und mit ihnen wachsen können.

**Transparente Zusammenhänge**

Durch die hohe Abstraktionsmöglichkeit bei der Anwendung objektorientierter Technologien kann eine bessere Verständlichkeit der globalen Zusammenhänge erreicht werden.

**Realitätsnahe Ergebnisse**

Durch den hohen Praxisbezug und die grosse Verständlichkeit können praxisgerechte und realitätsnahe Lösungen zur Verfügung gestellt werden.

**Höhere Qualität**

Die die Verwendung von bereits getesteten Komponenten können qualitativ stabilere Produkte abgeliefert werden.

**Bessere, schnellere Ergebnisse**

Durch die dynamischen Evolutionsmodelle kann auf Benutzerwünsche und geänderte Anforderungen über den gesamten Lebenszyklus besser und schneller reagiert werden.

**Grosse Wiederverwendbarkeit**

Komponenten stellen einen wesentlichen Teil der Wiederverwendbarkeit dar. Sie müssen deshalb so gebaut sein, dass si an möglichst vielen Stellen eingesetzt werden können. Durch die über die Zeit immer höhere Wiederverwendung kann Software gegenüber konventionellen Methoden in einem Bruchteil der Zeit und Kosten entwickelt werden.

## 1.3.3 Die Kosten

Objekttechnologie ist nicht gratis. Um die Vorteile vollumfänglich nutzen zu können, sind folgende Faktoren zu berücksichtigen:

1. **Investitionen** in neue Plattformen und Entwicklungsumgebungen, die diese Art von Softwareentwicklung unterstützen.
2. **Ausbildung** der Mitarbeiter in den neuen Methoden, Techniken und Werkzeugen.
3. **Schulung** der Geschäftsleitung, damit auch sie die neue Art des Denkens hinsichtlich der Softwareentwicklung versteht.
4. **Entwicklung** wiederverwendbarer Softwarekomponenten für zukünftige Projekte.

   **Die Investitionen zahlen sich auf Dauer aus!**