



Boris Wehrle

(E-Mail: [boris.wehrle@aitgmbh.de](mailto:boris.wehrle@aitgmbh.de))

ist Senior Consultant bei der AIT GmbH & Co. KG. Er berät Unternehmen bei der Softwareentwicklung auf Basis von Microsoft-Technologien. Seine Schwerpunkte liegen in der Konzeption von verteilten Anwendungen im industriellen Bereich sowie in der Unterstützung bei der Umsetzung von Entwicklungsprozessen auf Basis des Team Foundation Servers.

## Evolution einer Anwendung – ein Blick zurück für den Schritt nach vorn

Anwendungen unterliegen im Laufe ihres Lebenszyklusses einer Reihe von sich verändernden Rahmenbedingungen. Der Artikel beschreibt an einem Beispiel aus der Praxis die Weiterentwicklung der Architektur einer Anwendung – ausgehend von einer Einzelplatzinstallation, über ein Mehrbenutzersystem mit einer Auslagerung von unkritischen Teilbereichen in die Cloud bis hin zu einem vollständigen Cloud-Service. Dabei wird darauf eingegangen, welche Veränderungen zur Erreichung der jeweiligen Evolutionsstufe umgesetzt wurden, welche Probleme dabei auftraten und was man daraus lernen kann, um die Architektur einer neuen Anwendung zugleich einfach und zukunftsfähig zu gestalten.

Die Architektur einer Anwendung wird maßgeblich durch die Anforderungen an diese bestimmt. Im Rahmen des Anforderungsmanagements werden sowohl funktionale als auch nichtfunktionale Ziele erfasst. Auf Basis dieser Ziele, den technischen Möglichkeiten und den rechtlichen Rahmenbedingungen können unterschiedliche Architekturvarianten abgeleitet werden, die diese Ziele unterstützen.

Sowohl die Anforderungen wie auch die technischen Möglichkeiten und die Rahmenbedingungen unterliegen im Laufe des Lebenszyklusses einer Anwendung Veränderungen. Einige sind vielleicht unerwartet, manche liegen außerhalb des Einflussbereiches des Unternehmens, andere können mit entsprechendem Weitblick bereits vorweggenommen werden.

Eine der Aufgaben des Architekten ist es, die Anwendung über die aktuellen Anforderungen hinaus vorausschauend zu gestalten, um auf Veränderungen flexibel reagieren zu können. Dabei gilt es, die aktuelle Architektur einfach zu halten, um

die Komplexität und Dauer der Entwicklung zu reduzieren und zugleich Raum für Veränderungen zu lassen.

Bei der Konzeption einer neuen Anwendung lohnt sich neben der Prüfung aktueller Technologien auch ein Blick in die Vergangenheit und das Stellen folgender Fragen:

- Welche Konzepte haben sich bewährt?
- Mit welchen Veränderungen wurde man in den vergangenen Jahren konfrontiert?

- In welchen Bereichen verändern sich Technologien besonders häufig oder besonders stark?
- Gab es Veränderungen, die eine vollständige Neuentwicklung der Anwendung zur Folge hatten?
- Wurden Aspekte für zukünftige Erweiterungen implementiert, die nie zur Anwendung kamen?

Aus der Betrachtung der Architektur erfolgreicher oder auch gescheiterter Projekte kann man Rückschlüsse für zukünftige Entscheidungen gewinnen.

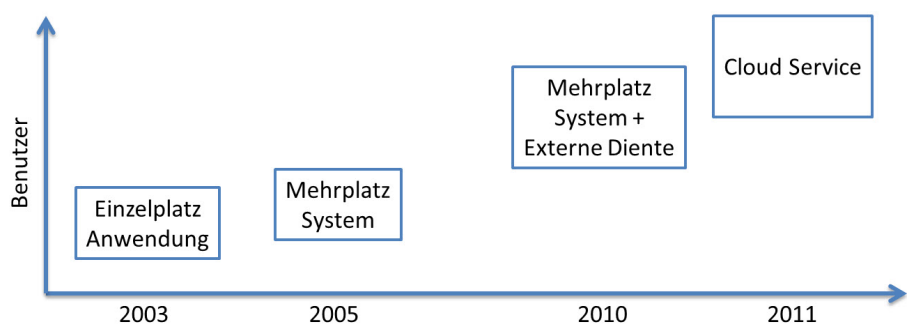


Abb. 1: Entwicklung der AIT-NetFactory.

Im Nachfolgenden lernen Sie die Geschichte einer industriellen Anwendung kennen, die von der AIT GmbH & Co KG seit dem Jahr 2002, wie in **Abbildung 1** dargestellt, kontinuierlich weiterentwickelt wurde.

Im Laufe des Lebenszyklus veränderten sich sowohl die funktionalen als auch die nichtfunktionalen Anforderungen. Folgen Sie der Entwicklung der AIT-NetFactory von einer Einzelplatzanwendung, über ein Mehrbenutzersystem bis hin zu einem Cloud-Service, um zu erfahren, wie mit wechselnden Technologien, stetig wachsenden Anforderungen an Skalierbarkeit, einer sich verändernden Infrastruktur sowie rechtlichen Unsicherheiten umgegangen wurde.

**Einzelplatzanwendung**

Die AIT-NetFactory ist ein Framework zur Steuerung und Visualisierung von Maschinen und Anlagen. Die erste darauf aufbauende Anwendung wurde in Fertigungsstraßen in der Glasindustrie eingesetzt. Der aktuelle Zustand der Anlage wird hierbei durch die Anwendung auf einem Leitstand visualisiert und ermöglicht es dem Bediener, Eingaben vorzunehmen. Die im Hintergrund gesammelten Daten können flexibel ausgewertet werden. Für die Bewältigung des Datenaufkommens und die Steuerung wurde die Anwendung auf einem Industrie-PC installiert, der unter Einbindung unterschiedlicher Hardwaresteuerungen die Aufgaben zuverlässig bewältigen konnte.

Im industriellen Umfeld steht bei der Technologieauswahl und Architekturkonzeption die langfristige Wartbarkeit im Fokus. Anlagen haben sehr häufig eine Nutzungsdauer von 15 bis 25 Jahren. Über diesen Zeitraum hinweg muss die Software gewartet und ggf. erweitert werden können. Ein Zeitraum, der in der IT-Industrie nicht überblickt werden kann. Die Veränderungen, die in dieser Zeit auf eine Anwendung zukommen, sind vielfältig und nicht vorhersehbar.

Entsprechend konservativ geht man in der Regel bei der Auswahl der Technologien vor. Bei der AIT-NetFactory wurde hiervon teilweise bewusst abgewichen. Zur Realisierung der Aufgabe sollten die jeweils aktuell leistungsstärksten Technologien verwendet werden, um zum einen eine zügige und ansprechende Visualisie-

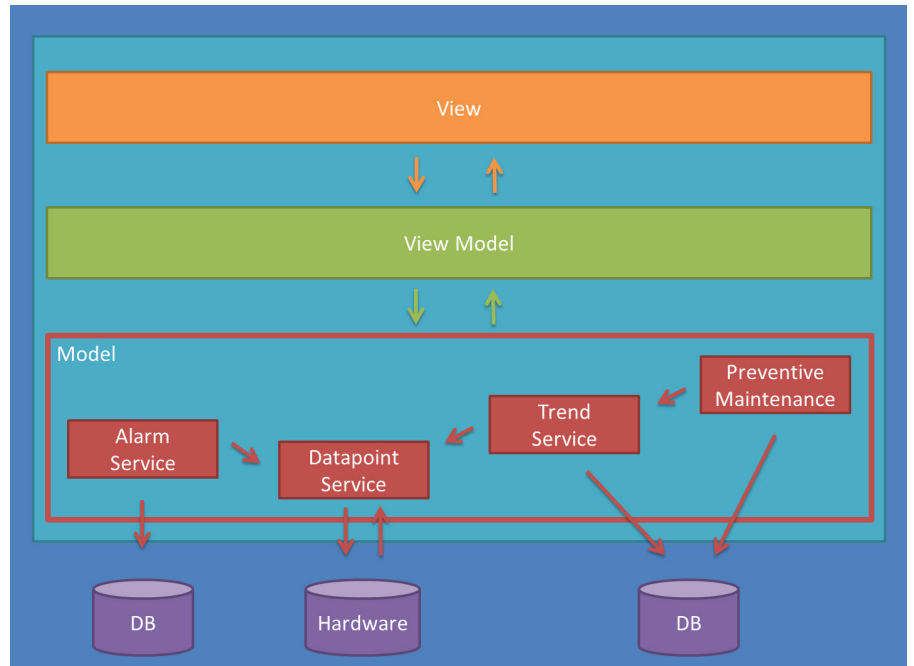


Abb. 2: Einzelplatzanwendung nach dem MVVM-Pattern

rung zu ermöglichen und zum anderen die Entwicklung in möglichst kurzer Zeit umzusetzen.

Bei der Auswahl der Basistechnologie wurde jedoch auf eine langfristige Unterstützung geachtet. Die Entscheidung fiel auf Microsoft.NET. Das damals noch recht junge Framework erschien vielversprechend. Es war in die langfristige Strategie von Microsoft eingebettet. Eine Unterstützung über Jahre hinaus wurde von Microsoft zugesichert [MPL].

Um das Risiko zu minimieren, wurde bei der Festlegung der Architektur darauf geachtet, einen flexiblen Austausch von Technologien und Komponenten zu ermöglichen. **Abbildung 2** zeigt, wie dies durch Einsatz eines dem Model-View-ViewModel (MVVM) vergleichbaren Pattern sowie einer Kapselung der Anwendungslogik in Services erreicht werden konnte.

Die Oberfläche der Anwendung basierte auf Windows Forms. Der Zugriff auf die Hardware sowie auf die SQL Server-Datenbank erfolgte durch die Services, die über Interfaces gekapselt wurden.

Bereits während der Entwicklung der ersten Version zeigte sich, dass die Oberfläche der Anwendung den stärksten Veränderungen, sowohl aus fachlicher als auch technischer Sicht, unterliegt. Die verwendeten Grafikbibliotheken entwickelten

sich schnell weiter. Neben neuen Funktionen brachten die Aktualisierungen leider auch häufig veränderte Schnittstellen mit sich, die regelmäßig nachgezogen werden mussten.

**Lernpunkte:**

- Bei der Wahl der Basistechnologie muss auf eine langfristige Ausrichtung geachtet werden.
- Eine lose Kopplung von Komponenten sowie die Verwendung von bewährten Patterns vereinfacht eine flexible Reaktion auf technologische Veränderungen.
- Mit der Anzahl der verwendeten externen Komponenten, wie z. B. Grafik- oder Kommunikationsbibliotheken, steigt die Häufigkeit von nicht beeinflussbaren wechselnden technologischen Rahmenbedingungen. Aus diesem Grund sollten externe Komponenten möglichst sparsam und ausschließlich in Bereichen eingesetzt werden, in denen sie problemlos ausgetauscht werden können.

**Mehrplatzsystem**

Die Anforderungen an die Leistungsfähigkeit des Systems stiegen stetig. Das Framework wurde für Anwendungen in immer größeren Anlagen eingesetzt. Die Aufgaben konnten schon bald nicht mehr durch einen einzelnen Rechner bewältigt werden.

Die räumliche Größe der Anlagen machte den Einsatz mehrerer Bedienterminals in Ergänzung zum Leitstand notwendig.

Um diese Anforderungen unterstützen zu können, wurde das ursprünglich als Windows-Anwendung konzipierte System in eine Client-Server-Architektur überführt.

Die bereits bestehende Entkopplung vereinfachte die Portierung. Die View- und ViewModel- Komponenten wurden, wie in **Abbildung 3** dargestellt, in einen Smart Client, der per ClickOnce auf die Bedienterminals verteilt werden konnte, integriert. Die Services wurden in einen Windows-Dienst, je nach Notwendigkeit in mehrere Dienste zur Verteilung auf unterschiedliche Rechner, ausgelagert. Die Kommunikation zwischen den Komponenten erfolgte per .NET Remoting.

Bei den ersten Lasttests zeigt sich, dass die verteilte Kommunikation sich negativ auf die Gesamtperformance auswirkte. Die bestehenden Schnittstellen machten den Einsatz von Remote-Objekten notwendig. Dies führte zu sehr häufigen entfernten Funktionsaufrufen mit sehr kleinen Datenmengen, worauf die eingeschränkte Gesamtleistung zurückzuführen war. Durch eine Umstellung der Schnittstellen auf eine nachrichtenbasierte Kommunikation konnte dies verbessert werden.

Später im Projektverlauf wurde .NET Remoting gegen die Windows Communication Foundation (WCF) ausgetauscht. Die Anpassungen konnte schnell vorgenommen werden, da bereits .NET Remoting aufgrund der Empfehlungen von Microsoft entsprechend dem WCF zugrunde liegenden Pattern eingesetzt wurde.

**Lernpunkte:**

- Bei der Konzeption der Architektur einer Anwendung sollten mögliche spätere Verteilungsszenarien geprüft werden. Kommt eine Verteilung in Betracht, sollte dies bereits in der ersten Version berücksichtigt werden. Der leicht erhöhte Entwicklungsaufwand in der ersten Phase wird bei einer späteren Verteilung mehr als kompensiert.
- Eine Berücksichtigung der Empfehlung des Herstellers beim Einsatz einer Komponente vereinfacht den Umstieg auf eine Folgetechnologie.

**Mehrplatzsystem unter Einbeziehung von Cloud-Services**

Der Bereich der Gebäudeautomatisierung stellt einen weiteren Einsatzbereich für das Framework dar. Die Anforderungen hierbei unterscheiden sich von den bestehenden Lösungen deutlich.

Während im Maschinenbau Anlage und Leitstand in der Regel räumlich nah beieinander liegen, ist in der Gebäudeautomatisierung auch eine Bedienung aus der Ferne, z. B. zur Temperaturvorwahl oder Verbrauchserfassung, interessant. Die dabei zur Anwendung kommenden Geräte können sehr vielfältig sein.

Bei der Analyse der Anforderungen zeichneten sich insbesondere in zwei Bereichen Probleme ab.

Die Oberfläche der Anwendung basierte inzwischen auf WPF. Die Technologie steht ausschließlich unter Microsoft Windows zur Verfügung. Eine Wiederverwendung auf mobilen Endgeräten ist damit ausgeschlossen. Bei genauerer Analyse der fachlichen Anforderungen stellte sich heraus, dass sich die Ziele der Anwendungen und damit die Inhalte der Oberflächen jedoch sehr stark unterschieden.

Während mithilfe der WPF-Anwendung zum Beispiel aufwendige Analysen erstellt wurden, war auf den mobilen Geräten nur ein Zugriff auf aktuelle Einstellungen und Kennzahlen notwendig. Damit erübrigte sich eine Portierung der bestehenden Oberflächen. Neue Bereiche wurden unter Rückgriff auf einen allgemeinen und plattformunabhängigen Standard (HTML 5 [W3C]) umgesetzt.

Ein weiteres Problem stellte der Zugriff der mobilen Clients auf die Anlagen und ihre Steuerungen dar. Um diesen zu ermöglichen, sind Eingriffe in die IT-Infrastruktur notwendig, was in den meisten Fällen aus Sicherheits- und Kostengründen nicht möglich ist. Eine Lösung bot die Verlagerung der relevanten Dienste in die Cloud (**siehe Abbildung 4**).

Eine Integration der Anlage wurde über den Windows Azure Service Bus gelöst. Dieser basiert auf gehaltenen ausgehenden Netzwerkverbindungen, die als Rückkanal verwendet werden. Eine Anpassung der Firewall ist hierfür im Regelfall nicht notwendig. Die mobilen Clients können dadurch direkt auf einen Cloud-basierten Dienst zugreifen.

Eine Verlagerung der WCF-basierten Dienste war ohne Probleme möglich. Um die Kompatibilität mit unterschiedlichen Clients zu verbessern, wurden Optimierungen vorgenommen. Neue Funktionen wur-

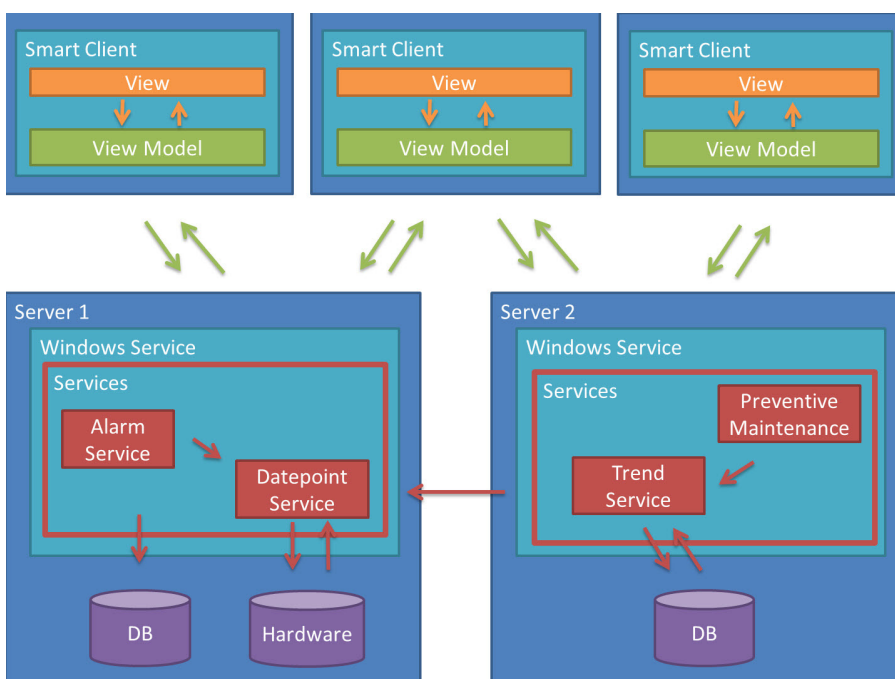


Abb. 3: Client-Server-Architektur

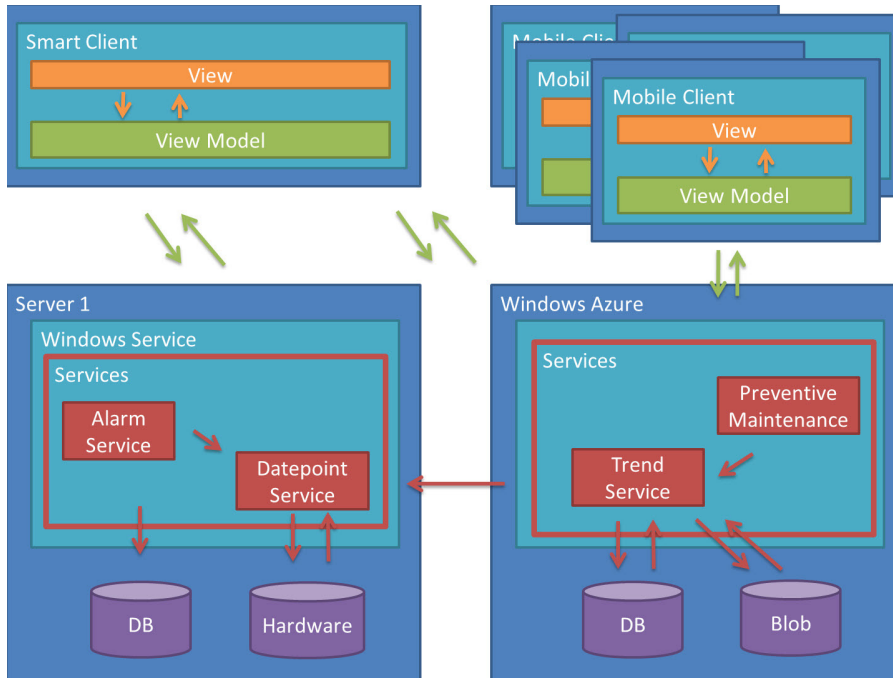


Abb. 4: Integration von Cloud-Services

den als REST-Dienste implementiert. Das Serialisierungsverfahren wurde auf JSON umgestellt.

Aufgrund der Kostenstruktur von Windows Azure wurde zusätzlich die Datenpersistierung angepasst. Durch die Verwendung von Dateien anstelle des SQL-Servers für Daten, die sich nur selten ändern, konnten die Betriebskosten weiter reduziert werden. Dies wirkte sich zusätzlich positiv auf die Geschwindigkeit der Anwendung aus.

**Lernpunkte:**

- Eine Verwendung von herstellerunabhängigen und anerkannten Standards erhöht die langfristige Wartbarkeit und die Reichweite einer Anwendung.
- Bei der Umsetzung von Anwendungen für unterschiedliche Endgeräte mit unterschiedlichen Aufgaben ist es häufig sinnvoll, eine für das Endgerät spezialisierte Technologie einzusetzen. Die Kosten für die Erstellung der einzelnen Anwendungen sind meist deutlich geringer als bei einer plattformunabhängigen Entwicklung. Die erstellten Anwendungen

fügen sich meist besser in das Bedienkonzept des Endgerätes ein und sind performanter.

- Eine Berücksichtigung in der Cloud kostenrelevanter Parameter bei der Entwicklung der Architektur einer Anwendung verbessert deren Performance auch bei einer ausschließlichen Verwendung in einem lokalen Netzwerk. Das Denken in monetären Größen anstelle von Kilobyte hilft dabei.

Für die Zukunft ist eine weitere Verlagerung von Funktionen der AIT-NetFactory in Windows Azure geplant. Hierdurch kann in vielen Fällen auf eine lokale Installation eines Servers verzichtet werden.

Steuerungsfunktionen werden von spezialisierten Hardwaregeräten übernommen. Rechenintensive Aufgaben sowie ergänzende Dienstleistungen werden unter Einbeziehung von Windows Azure realisiert. Die zusätzlichen Funktionen werden von einem Dienstleister als Service bereitgestellt. Der Betrieb der Anlagen wird damit einfacher und kostengünstiger.

Bei der Betrachtung von Windows Azure zeichnen sich Parallelen zu Microsoft .NET im Jahre 2002 ab. Die Tech-

nologie fügt sich in Microsofts langfristige Strategie ein. Eine Vielzahl von aktuellen und in der nahen Zukunft von Microsoft bereitgestellten Diensten basieren bereits auf dieser Technologie. Eine dauerhafte Unterstützung ist zu erwarten.

Dennoch ist auch hier eine Entkopplung sinnvoll. Alle ergänzenden Komponenten der AIT-NetFactory werden als Service entwickelt, der sowohl unter Windows Azure als auch auf einem lokalen Server betrieben werden kann. Einige der aktuellen Dienste können aufgrund offener juristischer Fragen noch nicht auf Windows Azure ausgelagert werden. Auf eine Klärung dieser kann durch den flexiblen Architekturansatz schnell reagiert werden.

**Lernpunkt:**

- Bei der Konzeption der Architektur einer Anwendung muss auf die Einhaltung von rechtlichen Rahmenbedingungen geachtet werden. Auf den Einsatz ggf. geeigneterer Technologien muss bei offenen Fragen oder Verstößen verzichtet werden.

**Fazit**

Ein Blick zurück hilft in vielen Fällen bei der Beantwortung aktueller Fragen. Insbesondere bei einer anstehenden Technologieauswahl ist es sinnvoll, Parallelen in der Vergangenheit zu suchen. Dies ermöglicht es zwischen kurzweiligen „Hypes“ und sich langfristig etablierenden Lösungen zu unterscheiden. Kombiniert man die getroffene Technologieauswahl mit einem Architekturansatz, der es ermöglicht, flexibel auf Veränderungen zu reagieren, kann man eine langfristige Wartbarkeit eines Produktes gewährleisten. ■

**Literaturverzeichnis**

[MPL] Microsoft Product Lifecycle Suche, <http://support.microsoft.com/lifecycle/search/>

[W3C] W3C, <http://w3.org>