Peter Maurer
(E-Mail: p.maurer@maurer-treutner.de)
berät seine Kunden aus dem Bereich der Entwicklung von Embedded Systemen seit vielen Jahren bei Fragestellungen zur Softwarearchitektur. Er ist seit 1995 Geschäftsführer und Mitinhaber der Maurer & Treutner GmbH & Co. KG.

Frameworks und Embedded Systems – ein Widerspruch?

Der Einsatz eines Frameworks ist eine architektonische Grundsatzentscheidung. Frameworks können die Entwicklungszeit für Embedded Software signifikant verkürzen, erhöhen jedoch oft den Ressourcenverbrauch der Systeme. Bei der Entwicklung von Embedded Systems ist deshalb eine sorgfältige Abwägung zwischen den Vor- und Nachteilen von Frameworks erforderlich.

Der Begriff „Embedded Systems“ umfasst eine weite Spanne sehr verschiedenartiger Systeme, die von Steuerungen auf Basis von 8-Bit-Controllern mit wenigen kByte Speicher in Geräten wie Waschmaschinen bis zu weitgehend auf moderner PC-Technologie beruhenden Systemen wie Set-Top-Boxen reicht. Dementsprechend vielfältig sind auch die Methoden und Programmiersprachen, die für die Entwicklung solcher Systeme zum Einsatz kommen.

Während bei der Entwicklung größerer Systeme Programmierumgebungen wie Java und C# beliebt sind, dominieren bei kleineren Systemen immer noch die Programmiersprachen C++ und C, insbesondere dann, wenn Echtzeitanforderungen erfüllt werden müssen.

Ein Framework besteht in der Regel aus einer Klassen- oder Komponentenbibliothek und einem Regelwerk für die Anwendungsarchitektur und die Programmierung. Anders als bei einer gewöhnlichen Klassenbibliothek findet eine Inversion of Control statt, d. h. der Kontrollfluss der Anwendung wird innerhalb des Frameworks gesteuert, eigener Code wird dazu beim Framework registriert und aus der Ablaufsteuerung des Frameworks heraus aufgerufen.

Ein Application Framework bringt die komplette Infrastruktur für die Anwendungen des Benutzers und somit eine bereits erprobte und bewährte Rahmenarchitektur mit. Die eigenen Anwendungen bewegen sich innerhalb dieses Rahmens – somit hat die Auswahl eines Frameworks weitreichende Auswirkungen auf die Architektur einer Anwendung und kann die Arbeit des Architekten ganz erheblich erleichtern.

Neben diesen architektonischen Aspekten bringen Framework-Bibliotheken in der Regel auch umfangreiche Lösungen von Standardproblemen mit und können so dazu beitragen, die Entwicklungszeit ganz erheblich zu verkürzen.

In der Entwicklung von Anwendungssoftware ist die Verwendung von Frameworks schon seit vielen Jahren etabliert und gehört dort quasi zum Standard. Im Gegensatz dazu war der Einsatz von Frameworks bei der Entwicklung von Embedded Systems lange Zeit kein Thema. Klassisch ist die Anzahl der Funktionen in einem Embedded System auf einige wenige, dafür aber hoch spezialisierte Steuerungs- und Regelungsaufgaben beschränkt.

Aufgrund der Speicherkapazität und der Rechenleistung ihrer Controller werden Embedded Systems bis heute oft so ausgelegt, dass die Aufgabenerfüllung der Software gerade eben gewährleistet ist. Die Komplexität solcher Systeme liegt dann in der Beherrschung der Echtzeitanforderungen und dem Umgang mit den sehr limitierten Ressourcen. An den Einsatz von Frameworks ist unter diesen Umständen kaum zu denken.

Durch die zunehmende Vernetzung von Systemen und die Entwicklung leistungsfähigerer Controller ist in vielen Bereichen die Zahl der funktionalen Anforderungen an Embedded Systems erheblich gestiegen. In – über Bussysteme oder Ethernet – vernetzten Systemen wird heute nicht nur der Zugriff auf Systemparameter erwartet, sondern auch die Funktionalität für Remote Updates und speziell im Consumer-Bereich die Konfiguration über HTTP/HTTPS.

So gehört mittlerweile ein Embedded Web-Server für viele Systeme zum „Standard“. Mit dieser Entwicklung geht natürlich eine erhebliche Steigerung der Komplexität der Embedded Software einher.

Hier kommen nun die Application Frameworks zum Tragen, denn sie können entscheidend dabei helfen, diese Komplexität zu beherrschen. Die Entwickler sind beim Einsatz eines Application Frameworks in der Lage, sich voll und ganz darauf zu konzentrieren, Lösungen für die speziellen Probleme ihrer Anwendung zu produzieren – für die allgemeinen Aufgaben müssen sie lediglich auf die Standardlösungen aus dem Framework zurückgreifen.

Viele Application Frameworks setzen auf einem Betriebssystem auf. Wenn das Board Support Package für das Betriebssystem vorhanden ist, kann somit auch das Framework auf der entsprechenden Hardware eingesetzt werden. Damit bringt die Entscheidung für ein Framework häufig auch gleich die Entscheidung für ein Embedded Betriebssystem mit sich.

Glücklicherweise stehen die meisten Frameworks für Implementierungen auf unterschiedlichen Betriebssystemen zur Verfügung. Deshalb kann die Verwendung eines Frameworks zur leichteren Portierbarkeit von Anwendungen führen, wenn in der Anwendung möglichst auf direkte Abhängigkeiten von Board Support Package (BSP) und Betriebssystem verzichtet wird. Im Idealfall kann sich eine Portierung dann auf die Anbindung des Betriebssystems an eine neue Hardware beschränken.

Embedded Frameworks werden entsprechend der Aufgaben von Embedded Systems für unterschiedliche Anwendungsbereiche angeboten und können auf unterschiedlichen Applikationsebenen ansetzen. Grundsätzlich ist es möglich, verschiedene Frameworks in einer Anwendung zu kombinieren, allerdings kann hier das Prinzip der Inversion of Control zu konzeptionellen Problemen führen, wenn gegenläufige Kontrollprinzipien angewendet werden.

Bei der Entwicklung von Embedded Systems kann die Verwendung eines Frameworks also helfen, die Entwicklungskosten und Entwicklungszeit zu verringern und gleichzeitig die Wartbarkeit des Produkts zu verbessern. Trotzdem muss hier eine sorgfältige Abwägung getroffen werden: Neben den erwähnten Vorteilen von Frameworks spielt nach wie vor ihr Ressourcenverbrauch eine nicht zu unterschätzende Rolle und zwingt die Entwickler von vornherein dazu, die Hardware entsprechend leistungsfähig auszulegen.

Gerade bei der Entwicklung von Produkten, die in kleineren oder mittleren Stückzahlen gefertigt werden, lohnt es sich oft, leistungsfähigere Controller und größere Speicherbausteine einzuplanen, um die eigentlichen Entwicklungskosten zu verringern – wie dargelegt, ist dies z. B. durch den Einsatz von Frameworks zu erreichen.

Ebenso kann sich der Einsatz von Frameworks lohnen, wenn ein relativ umfangreiches Produktportfolio entsteht, da der – durch das Framework bedingte – gleichartige Aufbau der Software den Entwicklern die Wartung der jeweiligen Produkte erleichtert.

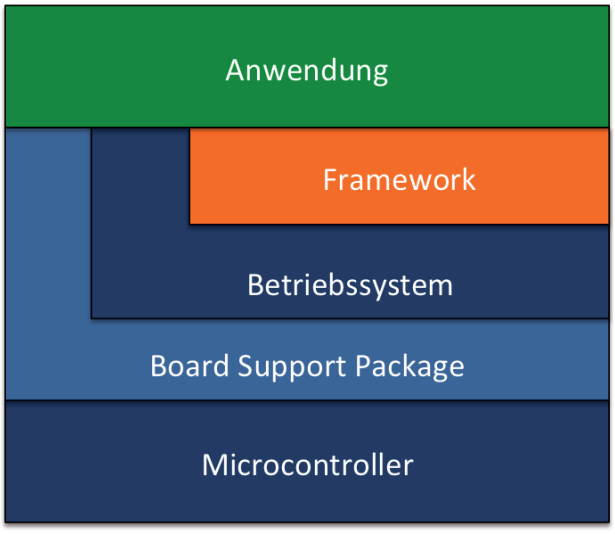

Abb. 1: Einbindung eines Frameworks in ein Embedded System

Ganz anders stellt sich die Situation dar, wenn große Stückzahlen einer Hardware produziert werden sollen – die Fixkosten der Entwicklung also klein gegenüber den variablen Produktionskosten sind. In diesen Fällen macht sich am Ende jeder auf der Hardware eingesparte Cent bemerkbar und das kann den durch den Verzicht auf ein Standard-Framework erhöhten Entwicklungsaufwand rasch aufwiegen. Trotzdem muss man auch in diesen Fällen nicht komplett auf die Vorteile von Frameworks verzichten: Architektonische Frameworks legen Regeln für den Aufbau von Software und Schnittstellen fest, überlassen aber die Implementierung dem Nutzer.

Beispiele für Embedded Frameworks

Die folgende Zusammenfassung erhebt keinen Anspruch auf Vollständigkeit, sondern stellt beispielhaft einige Embedded Frameworks vor:

AUTOSAR

Bei AUTOSAR (AUTomotive Open System ARchitecture) handelt es sich um ein architekturielles Framework für Steuergeräte in der Automobilindustrie, das den Austausch von Softwarekomponenten auf verschiedenen Steuergeräten erleichtern soll. AUTOSAR legt ausschließlich Standards und Schnittstellen fest, die Implementierung bleibt dem jeweiligen Hersteller überlassen.

Entsprechend ist AUTOSAR neutral gegenüber dem verwendeten Echtzeitbetriebssystem. AUTOSAR zielt insbesondere auf die Entwicklung von Komponenten in der Fahrzeugindustrie, von den in AUTOSAR verwendeten architekturiellen Konzepten können aber auch Anwendungen in anderen Domänen profitieren.

Poco-C++ / Open Service Platform

Bei OSP (Open Service Platform) handelt es sich um ein komponentbasiertes Application Framework, das auf den Open Source verfügbaren Poco-C++ Libraries aufbaut und besonders für den Einsatz in Systemen mit netzwerklastigen Aufgaben wie Settop-Boxen geeignet ist. Das Framework ist für Linux, Windows CE, QNX sowie einige Unix-Plattformen erhältlich.

OSP stellt eine Kombination aus einem Application-Framework und einer ziemlich horizontalen Sammlung von Libraries dar, die sich gut für Anwendungen wie Metering-Gateways eignet, die intensiv IP-basierte Kommunikationstechnologien, wie z. B. SOAP, verwenden.

QP™

Bei QP™ handelt es sich um ein sehr leichtgewichtiges State Machine Framework für C und C++. Dieses Open Source-Produkt stammt von Miro Samek und wird ausführlich in dessen Buch „Practical UML Stateharts“ beschrieben [Sam09]. QP™ liegt für eine ganze Reihe von Echtzeit-Betriebssystemen vor, es kann aber auch relativ leicht an noch nicht unterstützte Betriebssysteme angebunden und sogar auf Systemen ohne Betriebssystem eingesetzt werden.

QP™ eignet sich für den Einsatz in Systemen mit Echtzeitanforderungen und kann durch seinen geringen Ressourcenverbrauch auch in kleineren Systemen eingesetzt werden.

Fazit

Der Einsatz von Frameworks kann die Entwicklungszeit von Embedded Systems verkürzen und die Wartbarkeit der Software verbessern. Im Allgemeinen erhöht sich dadurch allerdings auch der Bedarf an Speicher und Prozessorleistung, sodass der Einsatz eines Frameworks erhebliche Auswirkungen nicht nur auf die Software- sondern auch auf die Hardwarearchitektur eines Systems hat. Deshalb ist eine sorgfältige Auswahl des Frameworks im Hinblick auf die Anforderungen aus der zu entwickelnden Anwendung erforderlich und sollte stattfinden, bevor die Hardwarearchitektur festgelegt ist. ■

Literatur und Links

[Sam09] Miro Samek : Practical UML Statecharts in C/C++, Second Edition, Elsevier, 2009

[Pas02] Alessandro Pasetti: Software Frameworks and Embedded Control Systems, Springer, 2002

[auto] http://www.autosar.com

[app] http://www.appinf.com/en/products/osp.html

[stm] http://www.state-machine.com/

[poc] http://www.pocomatic.com/