Embedded Software-Engineering-Kongress

Mit Rapid Embedded Prototyping Redesigns verhindern

Marco Schmid, Dipl. Ing. Systemtechnik FH, Schmid Elektronik AG, Schweiz

Ein neues Produkt wird entwickelt. Das erste Gerät steht auf dem Tisch. Nun stellt sich aber heraus, dass die Anforderungen unzureichend waren und deshalb das Produkt am Anwender vorbei entwickelt worden ist. Rapid Prototyping könnte solche Szenarien verhindern. Vor und während der eigentlichen Entwicklung wird im Schnellverfahren Hard- und Software entworfen, die dem Endprodukt in Form und Funktion sehr nahe kommen. Damit lassen sich frühzeitig wichtige Anforderungen validieren und Funktionen testen.

Bild 1: Am Beispiel eines Smart Meters wird gezeigt, wie Rapid Prototyping frühzeitig Anforderungen validieren und die Implementierung beschleunigen kann

Steigende Funktionalität und hohe Qualitätsanforderungen führen zu immer komplexerer Embedded Hard- und Software. Das treibt den Entwicklungsaufwand in die Höhe und doch stehen immer weniger Zeit, Geld und Personal zur Verfügung. Bietet sich zudem nur eine einzige Chance, das Produkt am Markt zu lancieren, müssen die Anforderungen stimmen und alle Designentscheide wollen abgesichert sein. Einen Weg aus diesem Dilemma bietet Rapid Embedded Prototyping, mit dem Ziel, Ideen im Bruchteil der üblichen Zeit umzusetzen und sofort zu testen – auf Formfaktoren, die dem Endprodukt nahe kommen oder sich sogar wiederverwenden lassen. Anhand der anschaulichen Prototypen lässt sich erstes Feedback und sogar das Vertrauen vom Kunden gewinnen.

Prototyp eines Smart Meters

Ein aktuelles Smart-Metering-Projekt soll als anschauliches Beispiel für die verschiedenen Aspekte von Rapid Prototyping zeigen (Bild 1). Es geht um den Bezug von elektrischer Energie gegen e-cash, was bei Waschmaschinen, Camping-Duschen oder "Tanken" von Elektroautos zur Anwendung kommt. Das System besteht aus zwei räumlich getrennten Modulen (Bild 2). Die Benutzerschnittstelle ist eine komfortable Bedieneinheit mit Touch-Display und einem RFID-Interface. Hier meldet sich der Kunde an, wählt eine freie Maschine aus, zahlt kontaktlos mit der RFID-Karte und startet den Prozess. Das Schalten der 230V-Speisung und die dreiphasige Messung der Ströme erfolgt in einer der bis zu vier "Powerboxen", welche mehrere Meter entfernt direkt bei den Maschinen installiert wird und mit der Bedieneinheit im Intervall von Millisekunden drahtlos kommuniziert.

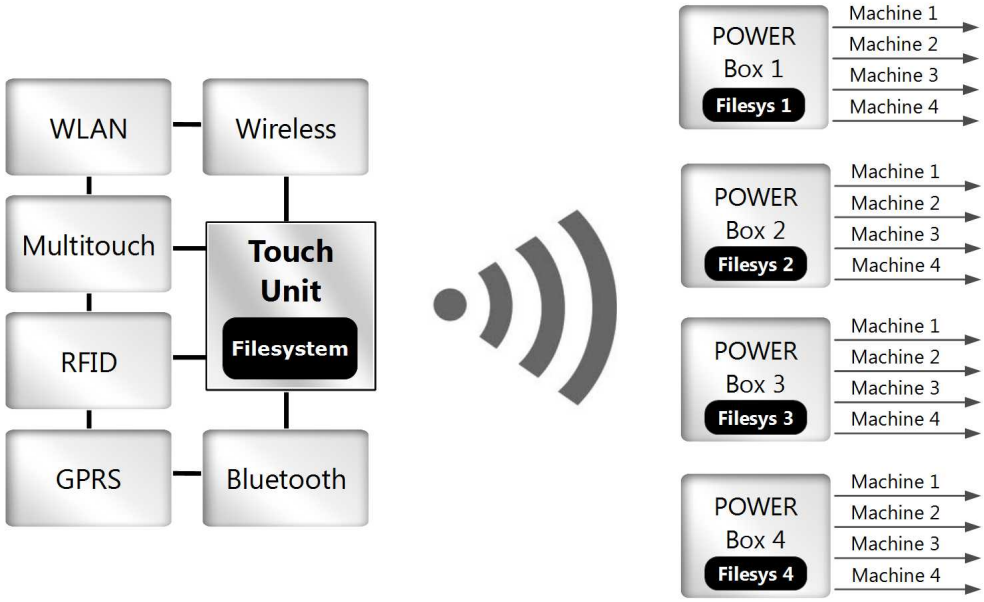

Bild 2: Das System besteht aus einer Touch-Unit (links) und mehreren Powerboxen (rechts), die drahtlos verbunden sind und elektronische Lasten schalten und messen

Warum Embedded Prototyping?

Solche Projekte werden normalerweise im bewährten „V"-Modell abgewickelt: Zuerst die Anforderungen definieren, die Spezifikation schreiben und implementieren. Danach verifizieren, ob die Spezifikationen eingehalten werden und ganz am Schluss validieren, ob das Produkt die Anforderungen erfüllt. Die grösste Schwäche dieser Methode ist ihre „Starrheit". Unterlaufen dem Designer während der Spezifikationsphase Fehler, verlängert sich zwangsläufig die Implementierungsphase und der Meilenstein wird überschritten. Im Fallbeispiel fehlte bei der gewählten Wireless- und RFID-Technologie die Felderfahrung. Ein Risiko, das mit grosser Wahrscheinlichkeit zu Verzögerungen geführt hätte. Schlimmer sind unzureichende oder falsch definierte Anforderungen, die im vorliegenden Projekt ein Redesign ausgelöst hätten.

Bei diesen zwei Problemzonen setzt Rapid Prototyping an, denn es erweitert das statische „V“-Modell um eine iterative, dynamische Komponente (Bild 3). Vor der zeitaufwändigen Design- und Implementierungsphase werden mit „Wegwerfprototypen“ die Anforderungen validiert. Während diesen zwei Phasen kommt anschliessend das „evolutionäre“ Prototyping zum Einsatz. Das ist ein praktisches Vehikel, um laufende Entscheide entwicklungsbegleitend abzusichern.

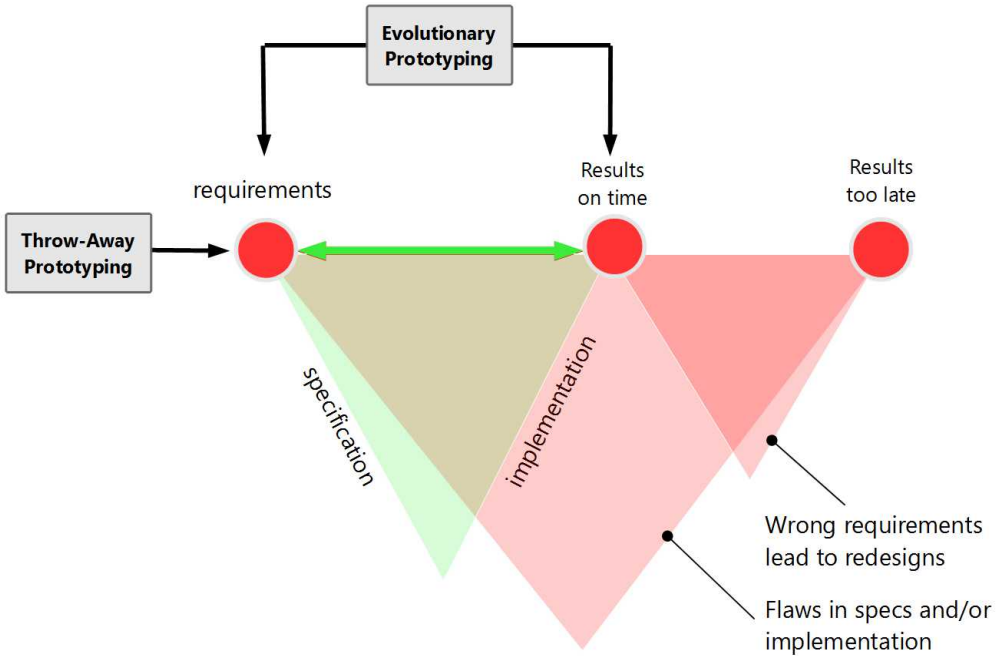

Bild 3: Die Vorteile zweier Welten nutzen: das qualitätssichernde «V»-Modell kombiniert mit agilem Rapid Prototyping von Embedded Soft- und Hardware

Für jeden Aspekt einen Prototypen

Ein Prototyp vermittelt den ersten Eindruck eines fertigen Systems und unterscheidet sich in Form und Funktion nur unwesentlich vom Endprodukt. Im Projekt können sich jedoch verschiedene Prototypen oft auf unterschiedliche Aspekte konzentrieren, zum Beispiel die Bedienerschnittstelle. Ausgehend von einem Zweizeilen-Charakterdisplay der ersten Gerätegeneration wagte der Hersteller des Smart Meters den Schritt zu einem grafischen Multitouch-Userinterface. Nach einer iterativen Entwicklung des Bedienkonzepts mit Bleistift und Papier wurde das User-Interface Schritt für Schritt und unter Miteinbezug der Endanwender auf einem Prototypen umgesetzt. Liegt ein funktionsfähiger Prototyp vor, sind die Anforderungen aus Anwendersicht geklärt, die kritischsten Designentscheide abgesichert und der Kern der Aufgabe gelöst. Alle am Projekt Beteiligten sind zu einem gemeinsamen Verständnis gekommen und die „richtige“ Entwicklung kann beginnen. Der Ausdruck „Rapid“ beim Prototyping ist deshalb so wichtig, weil es schnell verschiedene Varianten zu realisieren gibt und Unklares sofort ausgeregelt werden muss.

Entwicklungsmethode - Abstrakt und doch hardwarenah

Die wichtigsten sechs Anforderungen an die ideale Rapid Prototyping Entwicklungsumgebung für Embedded Systeme werden nachfolgend anhand der National Instruments Labview-Plattform diskutiert, sind aber lösungsneutral. Die erste Anforderung ist der schnelle Zugang zu neuer Technologie. So kann der Stand der Technik genutzt werden, ohne viel Zeit in die Lernkurve zu investieren. Zweitens muss das einfache Messen und Regeln über ein intuitives Prozess-I/O möglich sein, um die „trockenen“ Algorithmen frühzeitig mit echten Signalen zu verifizieren – ohne aufwändiges Entwickeln von Low-Level-Treibern. Als dritter Punkt ist wichtig, dass die Komplexität abstrahiert werden kann. Mathematische Operatoren sollen sich automatisch den vom Programmierer gewählten Datentypen (Skalar, Vektor, Matrix, zeitbasiertes Signal, Bitmap) anpassen. Die vierte Anforderung sind die Multitouch-Bedieneroberflächen, die den heutigen Anwendern aus ihrem privaten Umfeld von den Smartphones bekannt sind und in Zukunft auch bei Industrieanwendungen erwartet werden. Als nächster Punkt muss die Programmiersprache intuitiv und funktional sein (Bild 4). Traditionelle Elemente wie Datentypen, Strukturen, Arrays, Strings sollen ebenso unterstützt werden wie domänenspezifische Konstrukte (Regelungstechnik, digitale Filter), zustandsorientierte Programmlogik, Objektorientierung, modellbasiertes Design und textbasierte Algorithmen. In vielen Fällen ist die Sprache eine fachbezogene DSL (Domain Specific Language) und gehört der nächsten Generation (4GL, 4th Generation Language) an.

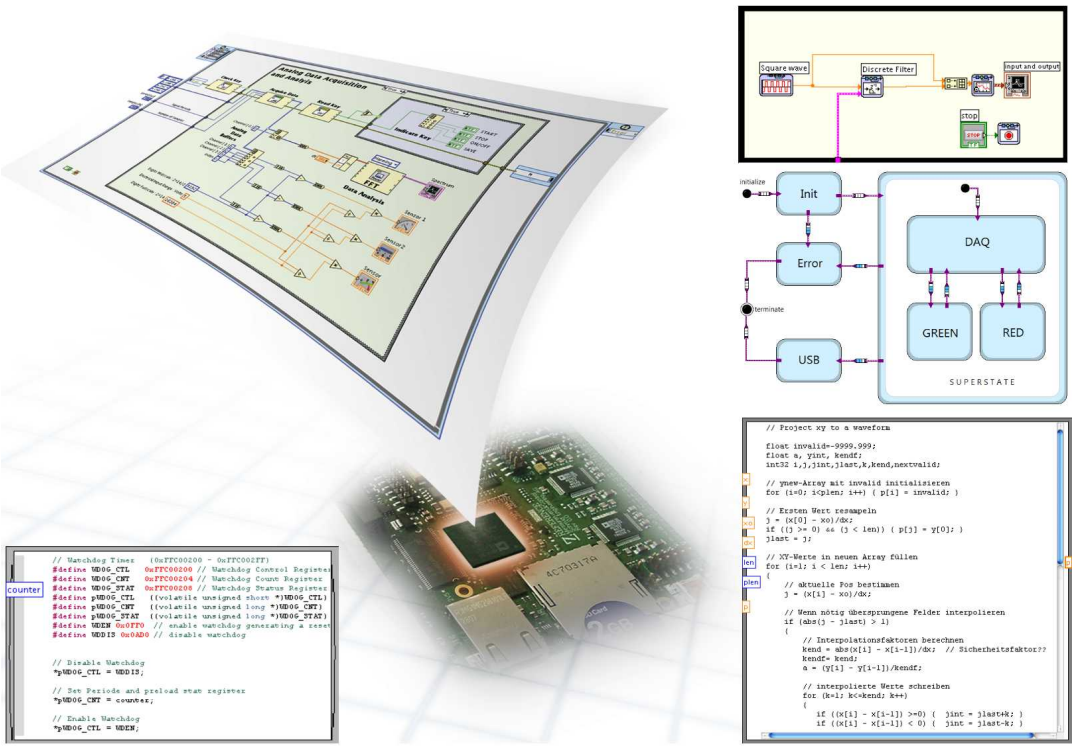

Bild 4: Eine intuitive, funktionale Programmierumgebung mit einer Auswahl an domänenspezifischen Konstrukten sorgt für schnelle Umsetzung einer Idee in die Realität. Alle Elemente werden im Systemblockschaltbild zusammengefasst.

Als letzte Anforderung muss eine Entwicklungsumgebung für Rapid Prototyping die volle Transparenz zum Mikroprozessor mit einem „C“-Interface, C-Code-Generator und schlankem Realtime-Kernel bieten. Damit steht die Türe für fast jeden 16/32-Bit-Microcontroller, Mikroprozessor und FPGA offen (Bild 5).

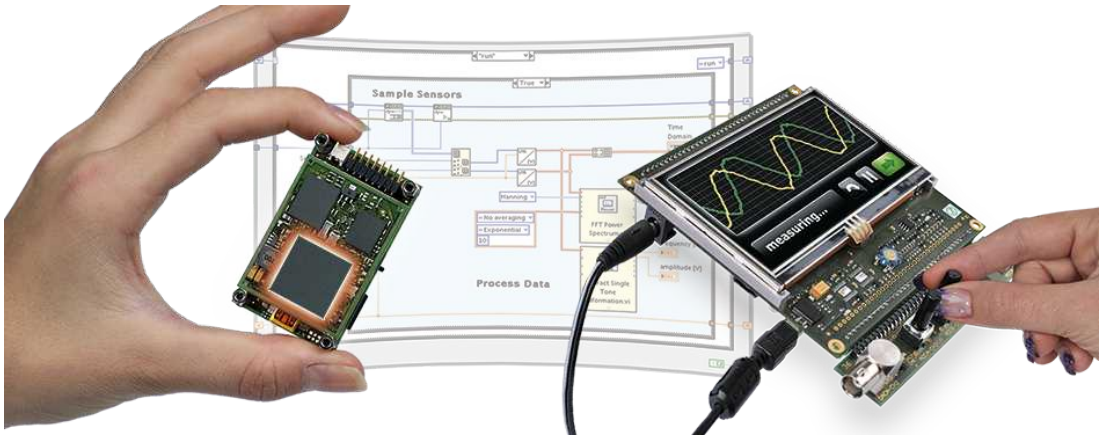

Bild 5: Mit effizienten Tools und wiederverwendbaren Hardware- und Firmware-Bibliotheken innerhalb weniger Wochen zu sofort einsetzbaren Hardwareprototypen

Die «Goldenen Regeln»

Die folgenden sechs Richtlinien für effizientes Rapid Prototyping halfen, das mit vielen Unsicherheiten und Risiken gespickte Smart-Metering-Projekt auf Kurs zu halten und erfolgreich zum Abschluss zu bringen:

Ideen kosten nichts. Das kreative Potential soll ausgeschöpft werden, denn die Frage, ob eine Idee wirtschaftlichen Nutzen hat, wird mit Prototyping schnell beantwortet: RFID anstelle Chipkarten, Farb-TFT plus Multitouch anstelle des Zweizeilen-Segmentanzeige, drahtlose Datenübertragung anstelle RS232, GSM/GPRS für die Fernwartung und Webserver als Konfigurationsschnittstelle. Ideen gab es genug - umgesetzt wurde gerade mal die Hälfte.

Das Rad nicht neu erfinden. Der Entwickler verfügte über wenig Erfahrung im Bereich Wireless/WLAN, RFID, Bluetooth und GSM/GPRS. Also evaluierte er fertige Funktionsmodule, die sich über UART und seriellem Kommunikations-protokoll sofort in den Prototyp einbinden liessen. Ein Mix aus Singleboard-computer, Plug-in-Coremodul und kundenspezifischer Hardware reduzierte die Entwicklungszeit auf ein Minimum.

Sich mehrere Wege offen halten. Alle Systemkomponenten wurden in der Software als Objekte implementiert. Das hat den Vorteil, dass sie für Varianten neu kombiniert werden können. Eine Zustandsmaschine sorgte für die Möglichkeit, neue Funktionalität unmittelbar einzufügen, ohne bestehenden Code zu ändern.

Ziel im Auge behalten. Für den Prototyp entwickelte Hardware, Software und Tools sollen sich im Serienprodukt grösstenteils wiederverwenden lassen. Deshalb wurde nicht auf exotische Technologie gesetzt, sondern im Hinblick auf Funktion und Leistung des in der Serie möglichen Mikroprozessors Bewährtes verwendet.

Schritt für Schritt zur Lösung. Zuerst wurden die Hauptfunktionen realisiert: Geld auf Karte transferieren, Mess- und Steuerdaten drahtlos kommunizieren und den Bediener grafisch durch den Prozess führen. Anschliessend wurde im zweiwöchigen Reviewintervall unter Miteinbezug des Auftraggebers fertigentwickelt.

Erzeugen von „Wow"-Effekten. Ein Prototyp soll nicht nur Brücken zwischen dem Stand der Technik und den Endanwendern schlagen, sondern etwas über das Ziel hinausschiessen und für Verblüffung sorgen. Vor allem der kontaktlose Transfer von e-cash kombiniert mit der intuitiven Multitouchbedienung ist in diesem Markt eine Innovation, die dem Kunden den entscheidenden Wettbewerbsvorteil bringt.

Qualität von Anfang an

Der Prototyp kommt dem Endprodukt sehr nahe, deshalb gilt der Qualität von Anfang an die volle Aufmerksamkeit. Im Beispiel mit dem Smart Meter war ein wichtiges Qualitätsmerkmal die drahtlose Kommunikation zwischen den zwei Boxen. Die Herausforderung bestand darin, einen robusten seriellen Datenaustausch aufrechtzuerhalten, damit sich die Bedieneinheit und die Leistungsbox nach temporären Stromausfällen, beispielsweise bei Sperrzeiten während der Mittagszeit durch das Elektrizitätswerk oder Übertragungsfehlern (z.B. Kollision mit anderen WLAN's), innerhalb weniger Sekunden selbständig wieder synchronisieren. Dies vermittelte dem Installateur ein komfortables Plug & Play-Gefühl. Aus der Sicht des Endanwenders „funktionierts einfach".

Lessons Learned ...

Richtig angewendetes Rapid Prototyping beschleunigt den Entwicklungsprozess von Embedded Systemen messbar und verbessert das Produkt spürbar. Der Miteinbezug und damit das Vertrauen der Endanwender steigt und die Kostenschätzungen sind präziser. Was das „Menschliche" betrifft, birgt Prototyping allerdings auch Risiken. Der Kontakt mit dem Kunden muss moderiert werden, sonst steigert sich die Erwartungshaltung ins Unermessliche. Ein Prototyp ist schliesslich kein Endprodukt, auch wenn es danach aussieht. Deshalb ist es wichtig, Rapid Prototyping erfahrenen Entwicklern zu überlassen, die neben Querdenken und Hartnäckigkeit auch über eine gesunde Portion Selbstvertrauen verfügen.

Referentenangaben

Marco Schmid, marco.schmid@schmid-elektronik.ch, (1969, Dipl. Ing. FH) ist Mitglied der GL/VR eines mittelständischen, global tätigen Industriebetriebes für Elektronikentwicklung und -produktion, grafisch programmierbare Mikroprozessorlösungen und Messgeräte für die Schiene. Sein Fokus liegt bei Hard- und Softwareentwicklung industrieller Embedded Systeme. Mit dem Hauptinteresse bei Software-engineering, DSL's / 4GL's, grafischer Abstraktion und der Brücke zur "Low-Level-Welt" mittels C-Code-Generatoren.