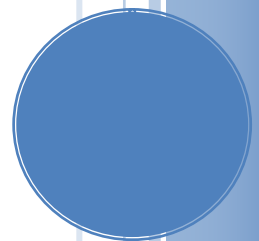


DSDM

Dynamic Systems Development Method

Laura Weber

08.05.2008



DSDM

Dynamic Systems Development Method

GLIEDERUNG

- I. Was ist DSDM überhaupt?
- II. Projektphasen und der DSDM-Prozess
 - a. Pre-Project Phase
 - b. Project Life-Cycle Phase
 - i. Feasibility Study
 - ii. Business Study
 - iii. Functional Model Iteration
 - iv. Design and Model Iteration
 - v. Implementation
 - c. Post-Project Phase
- III. 9 grundlegende Prinzipien und Voraussetzungen zur Anwendung von DSDM
- IV. Angewandte Kerntechniken
 - a. Timeboxing
 - b. MoSCoW
 - c. Prototyping
 - d. Testing
 - e. Configuration Management
- V. DSDM in der Praxis
 - a. Schlüsselrollen
 - b. Projektstruktur
 - c. Monitoring
- VI. Vorteile von DSDM
- VII. Referenzen

I. WAS IST DSDM ÜBERHAUPT?

DSDM (Dynamic Systems Development Method) ist eine agile Methode zur Softwareentwicklung, speziell für Informationssysteme (IS).

Ziel: Entwicklung einer Software, die die geschäftlichen Anforderungen bezüglich Zeit- und Kostenaufwand mit bestmöglicher Qualität erfüllt

DSDM ist ein auf RAD (Rapid Application Development) basierendes Kontroll-Framework zur Softwareentwicklung und wurde 1994 von einem gemeinnützigen Konsortium, bestehend aus Wirtschafts- und IS-Entwicklungsexperten entwickelt. Vertreter sind unter anderem Firmen wie British Airways, Hewlett Packard und Oracle. Die Mitglieder dieses Konsortiums kombinierten ihre best-practice-Erfahrungen und erstellten daraus eine Abhandlung von Prozessen und dazugehörigen Prinzipien, die die folgende Problematik klären sollten:

Warum funktionieren RAD-Projekte manchmal und manchmal nicht?

Hierin liegen die Wurzeln von DSDM begründet, aber auch heute wird das Framework noch kontinuierlich weiterentwickelt und vom Konsortium administriert.

DSDM ist eine Business-Value-Driven-Methode, was heißen soll, dass die Softwareentwicklung und somit auch deren Produkte immer darauf abzielt einen Geschäftswert zu schaffen. Hierbei spielt, wie in so vielen Bereichen von DSDM, vor allem die kontinuierliche Anwendereinbindung eine große Rolle.

Desweiteren bietet DSDM eine Vielzahl an Managementrichtlinien und –werkzeugen. Deren Fokus liegt vergleichbar mit Scrum, nicht auf der Entwicklungspraxis sondern eher auf dem Projekt- und Prozessmanagement. Jedoch lässt DSDM Raum für die Einführung solcher entwicklungsorientierten Methoden wie beispielsweise XP (in der aktuellsten öffentlich zugänglichen Version 4.2 ist XP standardmäßig bereits integriert).

Die wichtigsten mit einander korrespondierenden Größen bei DSDM sind

- Zeit
- Kosten
- Qualität

Wobei Zeit und Kosten zu Anfang des Projekts fix definiert werden. Die Qualität ist die unbestimmte Variable in der Gleichung. Angestrebt ist natürlich die bestmögliche Qualität, jedoch nur unter der Beachtung und Einhaltung der Zeit- und Budgetgrenzen.

„Lieber etwas früher ausliefern, das gut genug ist, als alles erst am Ende perfekt zu übergeben.“

Realtime-Solutins: <http://www.realtime-solutions.de/softwareentwicklung/dsdm/index.php>

II. PROJEKTPHASEN UND DER DSDM-PROZESS

a) Pre-Project Phase

Diese Phase stellt alle Beteiligten noch vor Projektbeginn vor die folgenden Aufgaben:

- Identifikation potenzieller Projekte
- Finanzielle (Projekt-) Absicherung
- Projektzustimmung

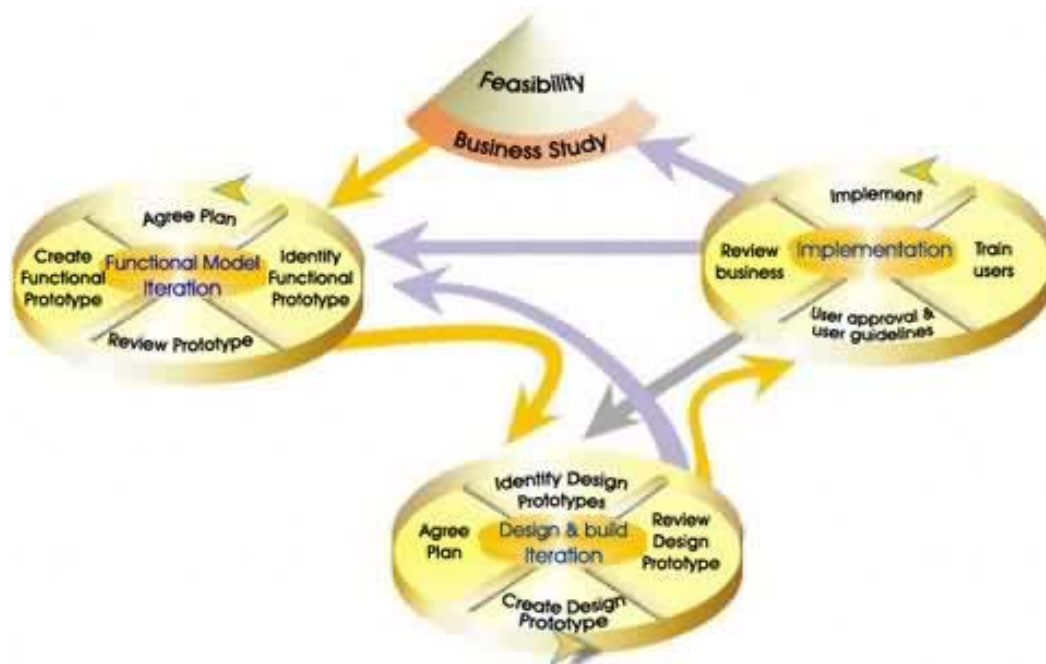
Um daraus resultierende Probleme, die in späteren Phasen des Projekts auftreten können zu vermeiden, sollten diese Punkte noch vor Projektbeginn gewissenhaft geprüft werden. Allerdings geht dies nicht einher mit einer zu detaillierten, sondern eher mit einer high-level-definierten Planung.

b) Project Life-Cycle Phase

Die Project Life-Cycle Phase stellt die aufwändigste Phase dar. Sie legt die Richtlinien und Vorgehensweisen fest, um ein IS in einem iterativen Prozess entwickeln zu können.

Die Feasibility Study (Machbarkeitsstudie) und die Business Study (Marktstudie) sind sequenzielle Phasen und ergänzen sich gegenseitig. Nach Abschluss dieser Studien, die nicht länger als einen Monat dauern sollen, wird das gewünschte IS iterativ und inkrementell in der Functional Model Iteration und Design and Build Iteration entwickelt. Ausgeliefert wird jeweils ein (fertiges) Teilprodukt, welches nachfolgend in der Implementation Phase beim Kunden eingeführt und getestet wird.

Ob hier die gleichen Teams für jede Anforderung nacheinander durch alle Phasen gehen, ob Teams parallel an verschiedenen Phasen arbeiten oder andere Formen der Arbeitsteilung gewählt werden, wird nicht festgelegt. Die Dauer einer Iteration beträgt jeweils zwei bis sechs Wochen.



http://www.xcess.nl/Portals/12/Expertise/DSDMproces_web.jpg

[Die gelben Pfeile verdeutlichen die sukzessive Reihenfolge der verschiedenen Phasen. Die blauen Pfeile wollen aufzeigen, dass zusätzlich immer wieder iterative und inkrementelle Schritte zur Korrektur und Verbesserung des bestehenden Systems möglich sind. Alle Änderungen in einem Projekt sind jedoch auch reversibel.]

i. Feasibility Study

Die Machbarkeitsstudie untersucht ein Projekt auf dessen Aussicht auf Erfolg bezüglich Projektdurchführung und Business Case.

Darüber hinaus wird die grundlegende Frage geklärt, ob sich ein Projekt überhaupt für den Einsatz von DSDM eignet. Als Indikatoren sind im Allgemeinen der Projekttyp, organisatorische und (zwischen-)menschliche Streitfragen zu nennen. Nicht zu vergessen ist die Berücksichtigung aller potenziellen Risiken bezüglich Technologie und Business Case.

Die Studie sollte kurz und prägnant sein und deren Ausarbeitung nur wenige Wochen dauern. Um dies zu erreichen werden in der Regel Workshops zwischen allen Projektbeteiligten arrangiert.

Der zu erstellende Machbarkeitsreport führt die aus der Praxis üblichen Themen in sehr geringer Detaillierung auf. Der Machbarkeitsprototyp, der optional entwickelt werden kann zeigt, ob ein Projekt und dessen Anforderungen überhaupt effizient umsetzbar sind.

Generell führt das Erstellen von Prototypen allerdings nicht unmittelbar zu einem inhaltlichen Mehrwert, da im Vorab schon gründliche Überlegungen bezüglich Business Case oder eben dem kritischen Objekt getätigt wurden. Der Prototyp kann in jeder Phase erweitert werden. Er stellt in erster Linie eine zusätzliche Absicherung von theoretischen Überlegungen dar und kann später bei positiver Resonanz der Stakeholder in das Projekt in originaler oder abgewandelter Form eingebunden werden.

Zusätzlich wird eine globale Rahmenplanung entworfen, die auch Erkenntnis darüber bringen soll, welchen Stellenwert die gewünschten Anforderungen haben und in welcher Relation der dafür benötigte Aufwand steht.

Der Risiko-Log hält die Risiken bezüglich der Umsetzbarkeit des Projekts in den jeweiligen Phasen fest und wird in jeder der folgenden Phasen aktualisiert.

Generell gilt für diese, als auch für die folgenden Phasen:

„Do enough but not more.“

Jennifer Stapleton

Produkte dieser Stufe:

- *Machbarkeitsreport*
- *Machbarkeitsprototyp*
- *Globale Rahmenplanung*
- *Entwicklungsplanung*
- *Risiko-Log*

ii. Business Study

In der Business Study werden der Fokus des Projekts, die wesentlichen Charakteristika von Business Case und den zu verwendenden Technologien ausgearbeitet, sowie die wesentlichen Projektbeteiligten und deren Verantwortungsbereiche festgelegt.

Es werden Workshops organisiert, in denen sich Kundenexperten zusammenfinden, um alle relevanten Fakten des Systems aufzulisten und mit Hilfe des MoSCoW-Prinzips zu bewerten. Die wichtigsten Businessprozesse und Bedürfnisse der verschiedenen Benutzergruppen werden identifiziert. Anhand dieser Geschäftsbereichsdefinition können später Entwicklungsprioritäten festgelegt werden. Um hierbei schon die Einhaltung von Zeit und Budget unter Beachtung der geforderten Qualität zu sichern setzt man die TimeBoxing-Methode ein (siehe *Kerntechniken*).

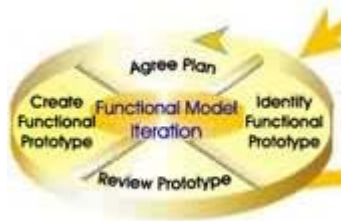
Die festzulegende Systemarchitektur stellt den Grundbaustein für die kommende IS-Entwicklung dar, bietet also eine initialisierte globale Architektur und muss daher optimal gewählt und definiert werden.

Desweiteren existiert wiederum die Option einen Prototyp zu erstellen und der Risiko-Log aus der Machbarkeitsstudie wird fortgeführt bzw. aktualisiert.

Produkte dieser Stufe:

- *Priorisierte Anforderungsliste*
- *Geschäftsbereichsdefinition*
- *Definition der Systemarchitektur*
- *Prototype*
- *Risiko-Log*

iii. Functional Model Iteration



In dieser Stufe wird eine detaillierte Spezifikation des Produkts festgelegt und die Systemarchitektur sowie ein Prototyp entwickelt. Dies erfolgt in 4 Unterstufen:

1) Identify Functional Prototype

- ***Anforderungsanalyse:***
Die Anforderungen an den aktuellen Prototypen werden, in Verbindung mit der bisher erstellten priorisierten Anforderungsliste, analysiert.
- ***Funktionale Anforderungsliste der aktuellen Iteration:***
Auswahl der funktionalen Anforderungen, die in den Prototype der aktuellen Iteration implementiert werden sollen und in den FUNKTIONALEN ANFORDERUNGEN aufgeführt werden.
- ***Liste der nicht-funktionalen Anforderungen:***
Liste der nicht-funktionalen Systemanforderungen, die in der NICHT-FUNKTIONALEN ANFORDERUNGSLISTE aufgeführt werden.
- ***Erstellung eines funktionalen Modells:***
Das Analysemodell und Prototype-Code werden als Sub-Aktivität eingebunden, um ein FUNKTIONALES MODELL zu entwickeln.

2) Agree Schedule

- ***Bestimmung der Zeit:***
Identifikation einer möglichen Zeittafel (TIMESLOT) um das Prototyping umsetzen zu können.
- ***Design-Entwicklung:***
Entwicklung des Prototyping-Plans, der alle nötigen Aktivitäten enthält, die in der verfügbaren Zeit zu erledigen sind
- ***Abstimmung:***
Ausarbeitung einer Abstimmungstabelle, wann und wie die Prototyping-Aktivitäten ausgeführt werden sollen.

3) Create Functional Prototype

- **Untersuchungen:**
Untersuchung der Anforderungen und Analysieren des FUNKTIONALEN MODELLS, welches in früheren Aktivitäten erstellt wurde.
Definition des IMPLEMENTIERUNGSPANS unter Berücksichtigung des Analysemodells als Grundlage für den Prototyp bei der nächste Sub-Aktivität.
- **Weiterentwicklung:**
Implementieren des FUNKTIONALEN MODELLS und des IMPLEMENTIERUNGSPANS, um einen FUNKTIONALEN PROTOTYPEN zu erzeugen. Dieser Prototyp kann weiterentwickelt werden, bevor er mit anderen Funktionen kombiniert wird. Er wird allmählich zu solcher Qualität geführt, dass es in das finale System eingebunden werden kann.
- **Konsolidierung:**
Konsolidierung des verfeinerten FUNKTIONALEN PROTOTYPS mit dem Prototyp aus früherer Iteration. Der neukombinierte Prototyp wird in der nächsten Stufe getestet.

4) Review Prototype

- **Testen des Prototyps:**
Der Prototyp wird auf Funktionalität und Lauffähigkeit geprüft.
- **Bewerten des Prototyps:**
Sammeln der Kommentare und Dokumentationen der Anwender.
Die Testberichte spielen eine wichtige Rolle bei der Erstellung des Bewertungsberichts. Auf Basis des FUNKTIONALEN PROTOTYPING REVIEW DOCUMENTS werden die priorisierte Anforderungsliste und der Risiko-Log aktualisiert. Nun wird über den weiteren Verlauf, ob eine zusätzliche Iteration vorgenommen werden soll, entschieden.

Produkte dieser Stufe:

- *Funktionales Modell*
- *Funktionaler Prototyp*

iv. Design and Build Iteration



In dieser Stufe wird das Produkt entwickelt und getestet. Dies erfolgt ebenfalls in 4 weiteren Unterstufen.

1) Identify Design Prototype

Identifizieren von funktionale und nicht-funktionale Anforderungen, die im System getestet werden müssen

2) Agree Schedule

Abstimmung, wie und wann diese Anforderungen umgesetzt werden.

3) Create Design Prototype

Erstellung eines Systems, das sicher an die Endnutzer für den täglichen Gebrauch und ebenso für Testzwecke ausgehändigt werden kann.

4) Review design Prototype

Überprüfung der Korrektheit des designten Systems. Testen und Bewerten sind wiederum die wichtigsten Techniken.

Produkte dieser Stufe:

- *Implementierungsstrategie*
- *Design Prototyp*
- *Nutzerdokumentation*
- *Testbericht*

v. Implementation



In dieser Stufe wird das fertige Produkt an die Anwender übergeben.

1) User Approval and Guidelines

Der Endnutzer genehmigt das getestete System für die Implementierung; Vorgaben bezüglich der Implementierung und Nutzung des Systems werden definiert.

2) Train Users

Die Endnutzer werden für die Nutzung des Systems trainiert.

3) Impelent

Implementierung des getesteten Systems auf den Maschinen der Endnutzer.

4) Review Business

Überprüfung des Einflusses, des in einem Unternehmen implementierten Systems:

Werden die Ziele, die zu Anfang des Projekts gesetzt wurden erreicht? Abhängig davon gelangt das Projekt auf die nächste Post-Project-Stufe oder muss nochmals iterativ weiterentwickelt werden.

Produkte dieser Stufe:

- *Übernahme*
- *Geübte Anwenderschicht*
- *Auslieferung des Systems*
- *Projektbewertungsdokument*

c) Post-Project Phase

Diese Phase sichert die effektive und effiziente Funktionalität des entwickelten IS. Diese basiert auf Instandhaltung, Verbesserungen und Korrekturen nach DSDM-Prinzipien.

Der Instandhaltungsprozess kann als kontinuierliche Weiterentwicklung auf Basis iterativer und inkrementeller Entwicklungsprinzipien von agilen Methoden gesehen werden. Anstelle eines Projektendes kann das Projekt wieder in vorhergehenden Phasen oder Stufen nachbearbeitet und verändert werden, sodass sich auch deren Produkte ändern können. Jede Änderung ist reversibel.

III. 9 GRUNDLEGENDE PRINZIPIEN UND VORAUSSETZUNGEN ZUR ANWENDUNG VON DSDM

Die folgenden 9 Prinzipien haben ihren Ursprung in den best-practice-Erfahrungen der Mitglieder des DSDM-Konsortiums.

1. Die Einbindung bzw. aktive Beteiligung der Anwender in die Arbeit des (Entwickler-) Teams ist unbedingt erforderlich.
2. Die Entscheidungsgewalt liegt überwiegend beim Team.
3. Im Mittelpunkt steht die regelmäßige Lieferung von fertigen (Teil-) Produkten
4. Abnahmekriterium: Business-Value-Driven Development!
Jede Lieferung muss einen Geschäftswert für den Kunden darstellen

Diese Prinzipien lassen sich direkt auf das agile Manifest abbilden. Sie stellen das Fundament der Methode dar. Weitere fünf Prinzipien bestimmen ihre Struktur:

5. Iterativ inkrementelle Entwicklung ist notwendig, um eine adäquate Lösung zu erzielen. (Durch frühen Produktiveinsatz der Neuentwicklungen und Einarbeiten von Anregungen und Verbesserungsvorschlägen in mehreren Iterationsstufen nähern sich die Projektpartner Schritt für Schritt der optimalen Lösung.)
6. Alle im Entwicklungsprozess vorgenommenen Änderungen sind umkehrbar.
7. Anforderungen werden nur grob auf einer hohen Ebene festgelegt und im späteren Projektverlauf konkretisiert.
8. Testen ist integraler Bestandteil des gesamten (Entwicklungs-) Prozesses.
9. Die kollaborative und kooperative Zusammenarbeit zwischen allen Beteiligten ist absolut wichtig.

Die Interaktivität zwischen Projektteam, zukünftigen Endnutzern und dem höheren Management muss gesichert werden mit dem Effekt, dass die Managementmotivation gesteigert und die Anwendereinbindung gesichert wird.

Die Zerlegbarkeit des Projekts in kleinere Teile muss ebenfalls garantiert und praktikabel sein. Sie vereinfacht eine iterative Behandlung und die Priorisierung von Teilaufgabe, welche oftmals bei ungenauer Einschätzung einen zeitlichen Aufschub des Projekts verursachen kann. Folgende, aus der DSDM-Praxis hergeleitete Annahmen unterstreichen nochmals die Wichtigkeit der Einhaltung der 9 Prinzipien:

- Kein System ist beim ersten Entwicklungsversuch perfekt.
(80/20-Klausel: 80% des Geschäftserfolgs kommen von 20% der Designanforderungen → Implementierung der ersten 20%, um die Endnutzer zufriedenzustellen, mit der Funktionalität und das die fehlenden 80% keine ernsthaften geschäftlichen Konsequenzen nach sich ziehen, dies lindert das Risiko von der Übertretung der Deadline und des Budgets.)
- Die Projektauslieferung sollte in angemessener Zeit, Budget und mit guter Qualität ausgeliefert werden.
- Jeder Entwicklungsschritt muss nur in soweit komplettiert sein, dass der nächste Schritt begonnen werden kann.

- Projektmanagement und Entwicklungstechniken sind miteinander verbunden.
- DSDM kann eingesetzt werden in neuen Projekten oder zur Erweiterung bestehender Projekte.
- Risikoabschätzungen sollten auf auszuführende Geschäftsfunktionen fokussiert werden, nicht auf den Entwicklungsprozess oder seine Werkzeuge
- Das Management legt mehr Wert auf Produktauslieferung als auf Aufgabenfertigstellung.
- Die Bewertung sollte auf der geschäftlichen Funktionalität liegen, nicht auf der Anzahl der lines of code.

Der Einsatz von DSDM ist nicht geeignet für...

... sicherheitskritische Projekte, da die hier benötigten ausführlichen Tests und Validierungen nicht mit den Zielsetzungen von DSDM übereinstimmen.

...Projekte, die auf wiederverwendbare Komponenten abzielen, da der Anspruch an Perfektion zu hoch ist.

IV. ANGEWANDTE KERNTECHNIKEN

a. TimeBoxing

Um die Ziele von DSDM zu erreichen ist TimeBoxing ein unverzichtbares essentielles Werkzeug. Es stellt eine Zeitmanagementtechnik in SWE-Projekten dar.

Das Projekt wird in einzelne Perioden aufgesplittet, in der Regel 2-6 Wochen. Jeder Teil erhält seine eigene Deadline, sein eigenes Budget und zu erledigende Teilaufgaben. Wird jeder Teil inkrementell in der vorgegebenen Zeit beendet, kann das Projekt erfolgreich mit mindestens zufriedenstellender Qualität durchgeführt werden.

Prinzip: Zeit und Budget sind fixe Größen

 Qualität / Arbeitsergebnisse sind variabel, können aber an die Bedürfnisse des Kunden angeglichen werden.

Die Teilaufgaben gehen darüber hinaus Hand in Hand mit der MoSCoW-Priorisierung. Der Fokus der Verantwortlichen liegt letztlich auf dem Erzeugen einer schnellen statt einer perfekten Lösung. Voraussetzung ist eine lückenlose Abfolge der Aufgaben und die Akzeptanz von möglicherweise unterschiedlicher Bearbeitungstiefe.

Eignung:

- Beschleunigung von Projekten allgemein, vor allem in früheren Projektphasen, bei denen sonst viel Zeit eingesetzt wird, da die Dringlichkeit noch nicht fühlbar ist
- Projekte, bei denen Zeitdruck schädlich ist (Grundlagenforschung, Design eines Sicherheitssystems für ein Kernkraftwerk)

Voraussetzungen:

- Verantwortliche müssen für die Dauer der geplanten Aufgabe 100% zur Verfügung stehen
- Angesetzte Zeitwerte müssen zumindest das Erstellen einer sinnvollen Teillösung erlauben
- Konsequenz, die sich durch das komplette Projekt zieht

Wirkungen:

- Dauerndes Gefühl der Dringlichkeit, kein Zeitverlust
- Fokus: generieren schneller Lösungen
- Anteil der inhaltlichen Arbeit steigt, da die Zeit fehlt, Ergebnisse perfekt darzustellen
- Vorgegebener Zeitplan wird eingehalten
- Dem Phänomen der Anfangsverzögerung in Projekten wird entgegengearbeitet und somit auch dem Verzug der Einhaltung der Deadline

b. MoSCoW

Die MoSCoW-Methode, stellt eine Technik dar, um die funktionalen Anforderungen des Kunden/Anwenders an eine Software besser priorisieren bzw. deren Wichtigkeit einstufen zu können. Es existieren 4 Priorisierungsstufen:

- MUST** -> hohe Priorität – **Minimum Usable SubseT**
Die hochpriorisierten Anforderungspunkte müssen unbedingt implementiert werden, da sie offensichtlich zur Steigerung des Geschäftswerts beitragen und unabdingbar sind. Sie müssen ausserdem in das TimeBoxing aufgenommen werden.
- SHOULD** -> mittlere Priorität
Der Geschäftserfolg hängt nicht von der Erfüllung dieser Anforderungen ab, aber es wäre vorteilhaft, diese zu erfüllen
SHOULDs sind oftmals genauso wichtig wie MUSTs, allerdings gibt es oftmals Umgehungsmöglichkeiten, um Anforderungen trotzdem zu erfüllen
- COULD** -> niedrige Priorität
Eine nice-to-have Anforderung, z.B. Tasks, die höhere Kundenzufriedenheit mit vergleichbar niedrigem finanziellen und zeitlichen Aufwand auslösen/erreichen.
- WON'T** -> wird momentan nicht benötigt, aber evtl **WOULD** in der Zukunft
Diese Anforderungen stellen völlig unkritische Elemente in der Projektentwicklung dar. Es sind meist weniger-ergiebige Teilbereiche, die in der momentanen Projektiteration nicht beachtet werden, aber evtl in späteren Reproduktionen zu berücksichtigen sind. -> **Would like to have in the future**

c. Prototyping

Eine Methode der Softwareentwicklung, die schnell zu ersten Ergebnissen führt und frühzeitiges Feedback bezüglich der Eignung eines Lösungsansatzes ermöglicht.

Sie garantiert eine gute Anwendereinbindung, Anforderungen können laufend präzisiert und verifiziert werden, dazu kommt noch eine frühzeitige Qualitätssicherung.

d. Testing

Diese Kerntechnik schreibt kontinuierliche Qualitätssicherung vor. In jeder Iteration der Entwicklungsphase muss von Entwickler und Anwender die Funktionalität der Software getestet werden. Jedes Team kann hierbei seine eigene Testmanagementmethode bestimmen.

e. Configuration Management

Das Konfigurationsmanagement erhält die Dynamik von DSDM, wenn während des Entwicklungsprozesses mehrere Dinge gleichzeitig abgehandelt werden müssen und die Produkte oftmals in einem hohen Tempo auszuliefern sind, denn nach Fertigstellung müssen sie sofort und streng kontrolliert werden.

V. DSDM IN DER PRAXIS

a. Schlüsselrollen

Im DSDM-Prozess werden verschiedene Rollen und Verantwortungsbereiche vergeben. Diese sind alle im DSDM-Manual festgelegt. Man unterscheidet zwischen Entwicklern und Anwendern.

Für jede Rolle ist eine gute Kommunikationsfähigkeit oberstes Gebot. Vor allem Entwickler benötigen eine gute Auffassungsgabe und eine für den Kunden verständliche, nicht zu sehr technisch versierte Ausdrucksweise.

Um die Kommunikationswege in einem DSDM-Team möglichst kurz zu halten, sollten die Teams nicht mehr als 2-6 Mitglieder haben. Wie viele Teams es insgesamt gibt hängt von der Komplexität und Größe des Projekts ab. Die Rollen der IT-Staff werden nicht explizit unterschieden, sie werden alle unter dem Begriff **Senior Developer** eingeordnet.

Schlüsselrolle bei den Entwicklern

- **Technical Coordinator**

Er ist für die Systemarchitektur, deren technische Umsetzbarkeit und Konsistenz verantwortlich. Zusätzlich kümmert er sich um die technischen Kontrollen und den effektiven Einsatz des Konfigurationsmanagements.

Anwender befinden sich traditionell nicht im DSDM-Team. Sie bringen ihr Wissen bei der Anforderungsspezifikation mit ein und überprüfen die aktuellsten Prozesse, indem sie Akzeptanztests durchführen. Die Anzahl der Nutzer, die hierfür eingebunden werden, können angemessen groß gewählt werden, um unter Beachtung aller Nutzersichten prüfen zu können.

Desweiteren können einige Anwender entweder als Teilzeitratgeber oder als Mitglieder im Projektteam eingesetzt werden.

Schlüsselrollen bei den Anwendern

- **Ambassador User**

Er ist der Anwendergruppe zugehörig und sein Verantwortungsbereich liegt in der Korrespondenz zwischen den Nutzern und der IT-Staff.

- **Adviser User**

Da die Rolle des Ambassador Users nicht immer ausreichend alle präsenten Gesichtspunkte abdecken kann, wird eine zusätzliche Rolle definiert. Er gilt als Interessent des Endsystems und kann aus IT-Staff, Administrator-Staff oder auch aus der Anwendergruppe sein. Er hat eine Ad-Hoc Anstellung, die wenn sie dem Projekt Nutzen bringen kann, aktiviert wird.

- **Visionary**

Er ist der Projektinitiator, jedoch nicht der Investor oder die oberste Entscheidungsinstanz in einem Projekt. Sein Aufgabenbereich liegt vor allem darin, in der Machbarkeits- und Businessstudie zu gewährleisten, dass die richtigen Entscheidungen bzgl. der Durchführung des Projekts getroffen werden. In späteren Stufen hat er die Aufgabe in wichtigen Präsentationen und Meetings darauf zu achten, dass das Entwicklerteam nicht die Sicht auf die essenziellen Business Objekte verliert.

b. Projektstrukturen

Die typische Größe für ein DSDM-Projekt umfasst 1-2 Teams mit maximal 6 Mitgliedern. Komplexere Projekte können in bis zu 6 Teams durchgeführt werden, die parallel arbeiten.

Existiert nicht mehr als ein Team, so ist der Technical Coordinator Bestandteil dessen. Bei mehreren Teams gibt es ebenfalls nur einen Coordinator, der aber nicht in einem Team integriert ist, sondern in einer Managerrolle agiert, die vor allem für die technischen Anweisungen und Kontrollen verantwortlich ist. Bei Bedarf besteht desweiteren die Möglichkeit diese Rolle aufzuteilen, beispielsweise in einen Systemarchitekten, eine technische Qualitätssicherung und einer Kontrollinstanz für Softwareentwicklung und Dokumentation.

Jedes Team sollte über Experten im Bereich Technik und Business verfügen, wenn nötig müssen zusätzliche Spezialisten im Rahmen eines Insourcing mit beratender Funktion hinzugefügt werden.

Werkzeuge zur Softwareentwicklung sollten ausreichenden Support garantieren können. Es ist auch förderlich, dem Team eine Tool-Einführung von einem Experten zu geben. Dadurch ersparen sich die Mitglieder viel Einarbeitungszeit.

Die Dokumentation erfolgt in Abhängigkeit vom Projekt nur schemenhaft. Die meisten Aspekte werden in direkter Kommunikation weitergegeben. Insgesamt definiert DSDM 15 Artefakte, für die aber weder Gliederung noch Inhalte vorgegeben werden. Statt dessen werden für jedes Artefakt vier Merkmale beschrieben:

- In welcher Phase es abgeschlossen wird,
- wer es abnehmen soll,
- welchen Zweck es hat und
- welche Qualitätskriterien für die Abnahme erfüllt sein sollen

c. Monitoring

Das Management muss nicht wie in traditionellen Projekten die Einhaltung von Zeitspannen überwachen, da dies durch das TimeBoxing bereits abgenommen wird, sondern die Integration und Qualität der nötigen „must-haves“.

Dies geschieht auf der Grundlage der priorisierten Anforderungsliste. Ein Gantt-Chart kann insoweit helfen, als dass es die TimeBoxes, Verantwortlichkeiten und parallelen Arbeitsprozesse übersichtlich darstellt. Was in den einzelnen Phasen genau produziert werden soll ist nicht aufgeführt.

Für eine detaillierte, auf arbeitsebene angelegte Überwachung und Kontrolle ist, wie sich erwies, das tägliche Meeting aller Teammitglieder unerlässlich. Dies soll kurz und prägnant sein und nicht länger als eine halbe Stunde dauern. Zusätzlich tragen Flurgespräche oder Stand-Up-Meetings zu schnellen Problemlösungen und vor allem zur Erhöhung der Kommunikation in der Softwareentwicklung bei.

VI. VORTEILE VON DSDM

- **Qualitätssteigerung** durch häufiges Feedback der Anwender, Reflexion, Inspektion und Anpassung
- **Produktivitätssteigerung** durch inkrementelle Emergenz von Anforderungen, Technologie und Team-Fähigkeit
- **Frühe Risikoerkennung**
- **Ergebnisse werden rascher geliefert** und entsprechen in der Regel den Erwartungen
- **Änderungen stellen kein Problem dar**
- **Zusammenarbeit im Team wird gefördert** durch Kollaboration und enge Kommunikation
- **Mitarbeiterfluktuation nimmt ab**, unter anderem durch die Ermächtigung der Selbstorganisation

Um die Synergieeffekte der agilen Softwareentwicklung nutzen zu können, müssen aus Management- und Entwicklersicht einige wichtige Punkte beachtet und umgesetzt werden:

Managementsicht	Entwicklersicht
<ul style="list-style-type: none"> ▪ Priorisierung der Anforderungen ▪ TimeBoxing ▪ Rascheres Feedback ▪ Häufige Inspektionen und Anpassung ▪ Offenheit und Vertrauen ▪ Einbindung der Stakeholder – Teampay 	<ul style="list-style-type: none"> ▪ Business-Value-Driven Development ▪ Coaching ▪ Emergenz ▪ Refactoring ▪ (Pair Programming)

Auch wenn DSDM zunächst sehr traditionell strukturiert zu sein scheint, zeigt es doch das typische Wertesystem agiler Verfahren – und bei näherem Hinsehen auch den notwendigen Spielraum für das Team. Das traditionelle Gewand hat aber auch Vorteile: DSDM ist als einziges agiles Verfahren bisher nach ISO 9001 zertifiziert und damit auch für solche Organisationen geeignet, bei denen das ein Ausschlusskriterium darstellt. Nachdem auch das Britische Verteidigungsministerium Mitglied im DSDM-Konsortium ist, bietet sich auch eine gute Argumentationsgrundlage für den Einsatz von DSDM im Umfeld von Behörden.

In Deutschland selbst ist, im Vergleich zu Großbritannien, der Einsatz von DSDM bisher noch wenig praktikabel. Dies könnte auch seinen Ursprung darin haben, dass das Konsortium für ein Unternehmen eine jährliche Lizenzierung zum derzeitigen Preis von rund 5000€ verlangt. Dieses Geld wird zwar zum Support und zur Weiterentwicklung von DSDM verwendet, ist allerdings ein nicht zu verachtender Punkt, vor allem weil andere agile Methoden wie XP und SCRUM kostenlos zur Verfügung stehen. Da sich die Tendenzen in der Zukunft aber stark auf die Verwendung von agiler Softwareentwicklung richten, lässt sich die Vermutung aussprechen, dass auch DSDM einen ordentlichen Aufschwung erfahren wird.

VII. REFERENZEN

- **Stapleton, Jennifer (1997):** DSDM – Dynamic Systems Development Method; Addison Wesley Longman
- **Stapleton, Jennifer (2002):** DSDM – Business Focused Development; 2. Auflage; Pearson Addison Wesley Prof.
- **Cockburn, Alistair (2006):** Agile Software Development – The Cooperative Game, 2. überarbeitete Auflage, Amsterdam, Addison-Wesley Longman
- **DSDM Consortium (1998):** Introduction DSDM into an Organization; In: www.dsdm.com
- **Quadrus Development Inc. (2003):** Crossing the agile chasm – DSDM as an Enterprise Friendly Wrapper for Agile Development; In: www.dsdm.com
- www.4managers.de/themen/time-boxing/
- www.legon.ch/sitelegon/sites/firma/dsdm.php
- www.denkwerft.de/cms/SEP/DSDM
- www.wikipedia.org
- www.realtime-solutions.de
- www.dsdm.com