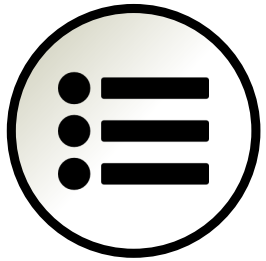


Gesetze der Software-Evolution

Systemtypen (nach Lehman, 1980)

S-System

P-System

E-System

Meir („Manny“) Lehman unterschied Softwaresysteme nach dem Grad ihrer Veränderungshäufigkeit in drei Typen.

Manny Lehman, 1925 - 2010

Systemtypen

S-System

P-System

E-System

S-System (“Wegwerf-System”)

- Nach Lehman: Entwickelt nach einer genauen **Spezifikation**, was das System leisten kann.
- Geht nach einmaliger Entwicklung in Betrieb.
- Keine Änderungen nach Inbetriebnahme, wenn das System nicht mehr passt: Wegwerfen
- Lebensdauer begrenzt, ca. 2 Jahre

Systemtypen

S-System

P-System

E-System

P-System (“Geplantes System”)

Nach Lehman: geschrieben, um feste Verfahren zu implementieren, die vollständig bestimmen, was das Programm tun kann.
Beispiel: Standardsoftware, etwa zur Lohnabrechnung
Überschaubare Änderungen, die Funktionalität ist vergleichsweise stabil (max. 10% Änderung im Jahr)
Es gibt regelmäßige Releases (z.B. 1x im Quartal)

Systemtypen

S-System

P-System

E-System

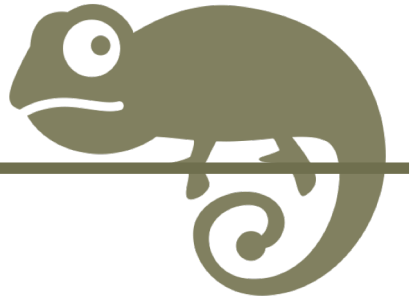
E-System (“Dynamisches System”)

Nach Lehman: bildet einen Teil der Realität ab. Wie das System sich verhalten soll, hängt stark von der Umgebung ab, in der es läuft, bzw. die es abbildet.
Muss sich kontinuierlich an Anforderungen und Umstände in dieser Umgebung anpassen.
Ständig neue Funktionen, Änderungen, nicht mehr benötige Funktionen – nie fertig, eher Produkt als Projekt.

Gesetze der Softwareevolution

Für die dynamischen Systeme (also den dritten Typ) beschreiben M. Lehman und L. Belady Gesetzmäßigkeiten, die sie bei großen Softwaresystemen beobachtet haben:

- Gesetz der fortdauernden Änderung
- Gesetz der zunehmenden Komplexität
- Gesetz der abnehmenden Qualität
- Gesetz der sinkenden Produktivität
- Gesetz des begrenzten Wachstums

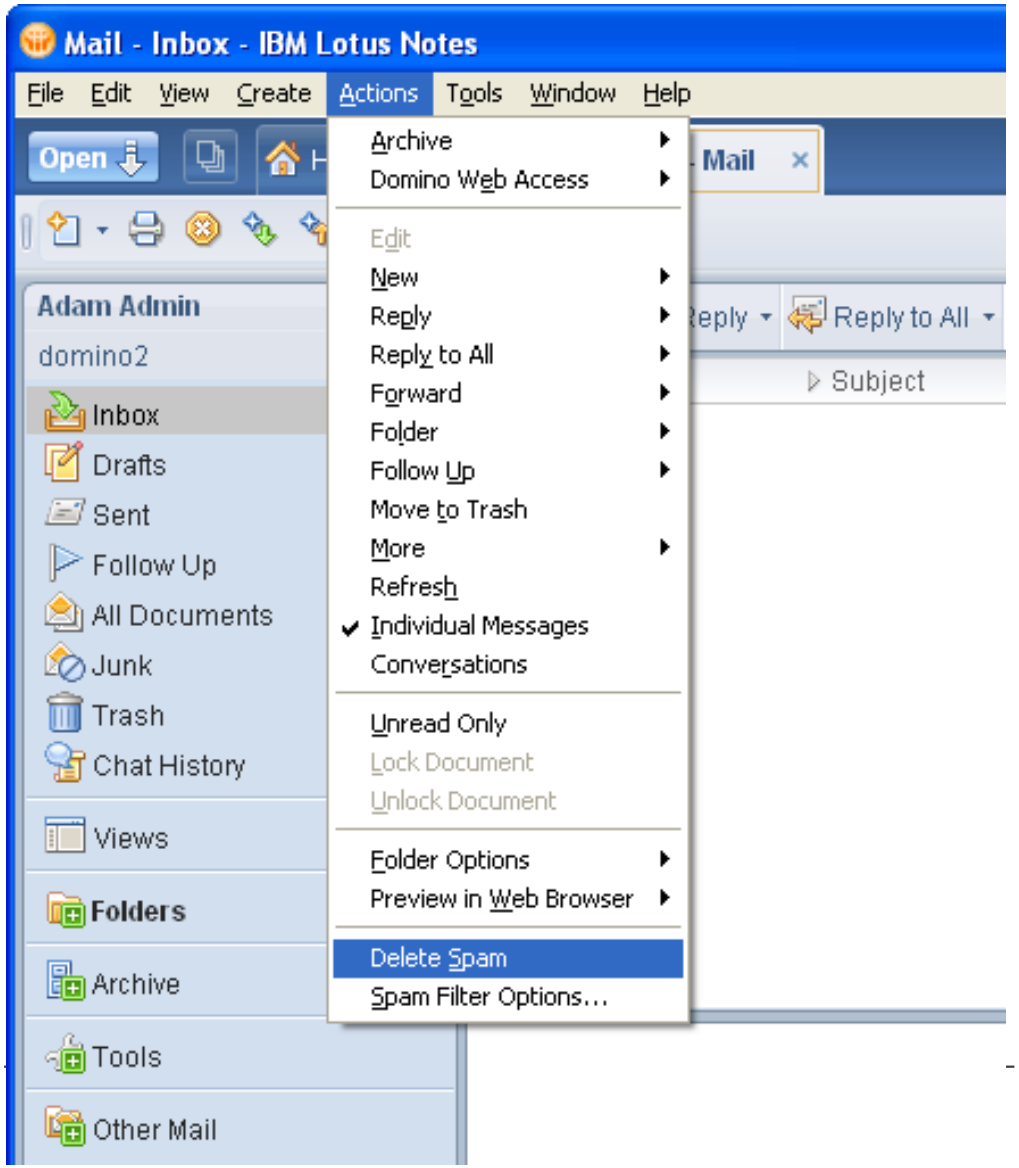


Fortdauernden Änderung

Gesetz der fortdauernden Änderung

- Nützliche Softwaresysteme modellieren Abläufe und Objekte der realen Welt.
- Wenn die Abläufe und Objekte sich ändern, muss die Software nachziehen.
- Software, die stehen bleibt oder sich zu langsam anpasst, wird zunehmend nutzloser.

➔ Der Besitzer der Software ist gezwungen, sie ständig weiterzuentwickeln, wenn er deren Nutzen erhalten will.


Mail - Inbox - IBM Lotus Notes
File Edit View Create Actions Tools Window Help
Open
Mail
Archive
Domino Web Access
Edit
New
Reply
Reply to All
Forward
Folder
Follow Up
Move to Trash
More
Refresh
Individual Messages
Conversations
Unread Only
Lock Document
Unlock Document
Folder Options
Preview in Web Browser
Delete Spam
Spam Filter Options...
Reply
Reply to All
Subject
Adam Admin
domino2
Inbox
Drafts
Sent
Follow Up
All Documents
Junk
Trash
Chat History
Views
Folders
Archive
Tools
Other Mail

Zunehmende Komplexität

Gesetz der zunehmenden Komplexität

- Zu Beginn orientiert sich die Struktur der Software (Zerlegung, Abhängigkeiten) an der geforderten Funktionalität.
- Änderungen und Ergänzungen in der Funktionalität führen zu neuen Teilen und Beziehungen, die Komplexität steigt.

➔ Die (ursprüngliche) Zerlegung folgt später nicht mehr der geforderten Funktionalität (Fachlichkeit).

Abnehmende Qualität

Gesetz der abnehmenden Qualität

- Es entstehen immer mehr Widersprüche zwischen den neuen Anforderungen und der bestehenden Struktur der Software.
- Neue Systemeigenschaften werden auf Kosten der Stimmigkeit gekauft.

➔ Die Fehler nehmen zu und die Wartbarkeit nimmt ab. Die Qualität sinkt.

Sinkende Produktivität

Gesetz der sinkenden Produktivität

- Durch gestiegene Komplexität und geringere Qualität erhöht sich der Aufwand für Änderungen und Erweiterungen.
- Vor einer Änderung sind mitunter umfangreiche Nachbesserungen erforderlich.
- Pathologische Phänomene im Design (Starrheit, Zerbrechlichkeit, ...) kosten zusätzlichen Aufwand.

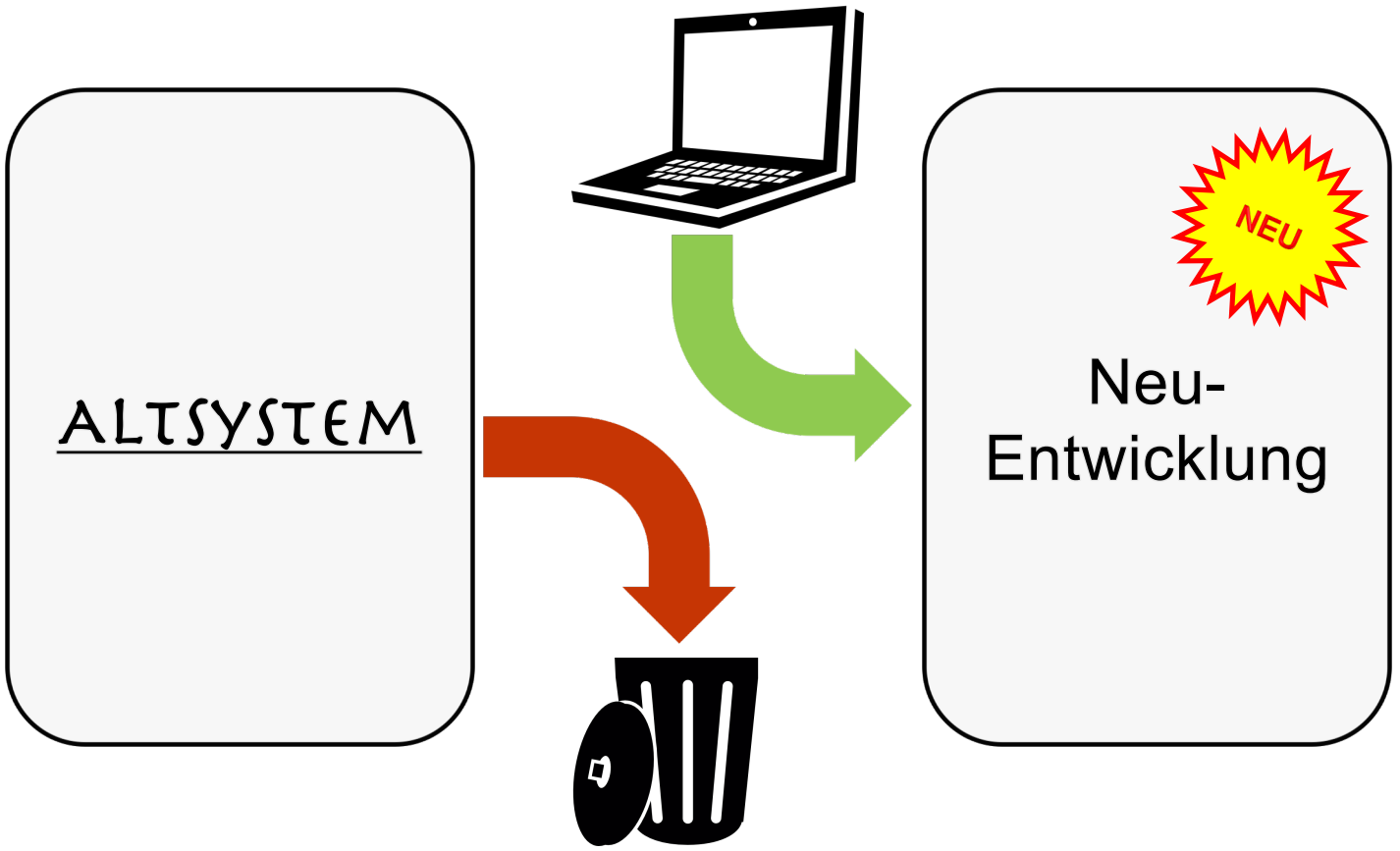
➔ Jede zusätzliche Änderung oder Erweiterung wird teurer.

Begrenztes Wachstum

Gesetz des begrenzten Wachstums

- Anforderer und Anwender müssen wegen abnehmender Produktivität länger auf neue Funktionalität warten, und mehr investieren.
- Die Akzeptanz für das System sinkt. Es wird begonnen sich nach Alternativen umgesehen.
- Die Weiterentwicklung kommt zum Stillstand.

➔ Das System wird ersetzt.


ALTSYSTEM
NEU
Neu-
Entwicklung