„Cut-And-Paste Programming“

„Golden Hammer“

Christian Stahl

01.02.2002

Gliederung

1. Einführung
2. Symptome
3. Konsequenzen
4. Ursachen
5. Lösung

„Cut-And-Paste Programming“

Einführung

- auch bekannt unter:
  - „Clipboard Coding“
  - „Software Cloning“
  - „Software Propagation“
- Ursache: Faulheit

Einführung

- Anekdote:

  „Ich dachte, du hast den Fehler behoben. Warum ist er wieder aufgetreten?“

  „Ihr arbeitet schnell. Über 400.000 Quelltextzeilen in 3 Wochen ist wirklich eine gute Leistung!“

Einführung

- Beispiel:
  - verkettete Liste, die im Quelltext mehrmals vorkommt
  - Speicherrückgabe vergessen
  - Fehler wird gefunden, verbessert und per „Cut-And-Paste“ im Quelltext eingefügt
  - Fehler tritt trotzdem noch auf, obwohl das verbesserte Quelltextstück richtig ist

Einführung

- Wiederverwendung von Quelltext durch Kopieren von Source-Teilen
- leichter Quelltext zu modifizieren als neu zu schreiben
- positive kurzzeitige Konsequenz:
  - Quelltextzeilen nehmen zu
  - Veränderungen sind leicht einzufügen

Symptome

- über 10000 Quelltextzeilen pro Woche
- viele identische Änderungen, um ein Problem zu beheben
- derselbe Fehler kommt wieder vor

Konsequenzen

- viele einmalige Fehlerbehandlungen und Tests für denselben Quelltext entwickeln
- übermässige Wartungskosten
- Anwachsen der Quelltextzeilen, ohne die Wartungskosten zu reduzieren

Ursachen

- kurzfristige Ausgaben wichtiger als die langfristige Investition in wiederverwendbaren Quelltext
- Inhalt des Moduls nicht geschützt
- Entwicklungsgeschwindigkeit überschattet alle anderen Faktoren
- wiederverwendbare Komponenten sind schlecht dokumentiert und nicht schnell verfügbar

Ursachen

- schlechtes Verständnis von
  - Vererbung
  - Komposition
  - andere Entwicklungsstrategien
- Unerfahrenheit mit neuer Technologie
  - modifizieren eines existierenden, lauffähigen Beispiels

Lösung

- Black-Box-Wiederverwendung:
  - basiert auf Objektkomposition (Zusammensammeln von Objekten, um komplexeres Verhalten zu erhalten)
  - zusammengesetzte Objekte zeigen einander keine internen Details
  - gleichen somit Black-Boxes

Lösung

- Black-Box-Wiederverwendung bei grossen Systemen
  - Interface kann vom Klienten nicht verändert werden
  - Implementation des Objektes vom Interface unabhängig machen (z.B. mit Hilfe einer IDL)

Lösung

- beseitigen des Cut-And-Paste:
- aus Basiscode eine Bibliothek oder Komponente machen (Black-Box-Prinzip)
- Zerlegungsprozess kann schwierig, lang und teuer sein

1. finden der kopierten Quelltextstücke
2. Standardversion entwickeln
3. zukünftiges Vorkommen verhindern (prüfen des Quelltextes)

Lösung

- Black-Box-Wiederverwendung reduziert
  - Wartungskosten durch gemeinsamen Quelltext
  - Tests
  - Dokumentation
- Verwandte Lösungen: „Spaghetti Code“
  - enthält oft mehrere Instanzen des Cut-And-Paste Antipattern
  - Cut-And-Paste oft einzige Möglichkeit Teile des Quelltextes wiederzuverwenden

„Golden Hammer“

Einführung

- Kompetenz eines Entwicklungsteams im Umgang mit einem Produkt bzw. einer Technologie
- jedes neue Produkt scheint mit dieser Technologie bestens entwickelt zu werden
  - Hammer meistens eine Fehlanpassung
  - zu wenig nach alternativen Lösungen gesucht
- Argumentation: zukünftige Erweiterungen des Produktes bei Verwendung des Hammers minimieren Risiko und Kosten

Einführung

- auch bekannt unter:
  - “Old Yeller”
  - “Head-in-the sand“
- Ursache:
  - Ignoranz
  - Stolz
  - Engstirnigkeit

Einführung

- Beispiel:
  - Call-Center-System
  - MS Access als Dauerlösung für Client/Server-Anwendung
  - Benutzeroberfläche auf Access aufgesetzt
  - durch diese schlechte Architektur war das System von Access abhängig
  - scheiterte nach 6 Monaten

Einführung

- Anekdoten:

  „Ich habe einen Hammer und alles andere ist ein Nagel.“

  „Unsere Datenbank ist unsere Architektur.“

  „Vielleicht hätten wir doch nicht Excel-Makros verwenden sollen.“

Symptome

- identische Technologie für konzeptuell verschiedene Probleme
- Anforderungen auf das zugeschnitten, was die Technologie gut kann
- Entwickler isoliert, mangelndes Wissen und Erfahrung mit Alternativen

Konsequenzen

- Lösungen haben minderwertige Leistung, Skalierbarkeit im Vergleich zu anderen Lösungen in der Industrie
- neue Entwicklung hängt stark von spezifischem Produkt oder Technologie ab
- System-Architektur durch Hammer beschrieben

Ursachen

- Entwicklungsteam ist mit einer Technologie hochprofitabel
- verschiedene Erfolge mit der Technologie
- Technologie und Sammeln von Erfahrung damit war grosse Investition
- Entwicklungsteam kommt mit der Industrie bzw. anderen Firmen nicht in Berührung
- Abhängigkeit von patentierten Produktmerkmalen, die bei anderen Produkten nicht sofort verfügbar sind

Bekannte Ausnahmen

- Produkt ist Teil einer Produktgruppe
- Hammer ist strategische Langzeitlösung (Oracle DB)

Lösung

- Firma muss die Erforschung neuer Technologien festlegen
- professionelle Schulung von Mitarbeitern
  - Forschungstreffen
  - Buchstudiengruppen
  - Konferenzen
  - Schulungen
  - Gratifikationen
- offene Systeme und Architekturen benutzen

Lösung

- Einstellen von Mitarbeitern mit unterschiedlichen Kenntnissen
- definierte Schnittstellen, um den Austausch von Subsystemen zu vereinfachen

  → Austausch der Implementation, ohne Einfluss auf die anderen Komponenten
- Standardschnittstellen implementieren (CORBA, TCP/IP)

Lösung

- Verwandte Lösungen:
  - „Lava Flow“
    - Entfernen von „totem“ Quelltext
    - Quelltext von alten Versionen des Hammers werden ungern modifiziert
  - „Vendor Lock-In“
    - Entwickler werden vom Hersteller unterstützt den Hammer anzuwenden
    - Software Projekt soll von einem Hersteller abhängen

Vielen Dank für die Aufmerksamkeit!

Fragen?