

Analyse von Versionshistorien

“Programmers who changed these functions also changed ...”

Mark Wiesemann

Seminar „Software Maintenance“

28. März 2006



Übersicht

- 1 Einführung
- 2 eROSE
- 3 Weitere Ansätze
- 4 Zusammenfassung

Übersicht

- 1 Einführung
- 2 eROSE
- 3 Weitere Ansätze
- 4 Zusammenfassung

Versionshistorie

Versionshistorie ...

- wird bei der Softwareentwicklung verwendet
- verwaltet verschiedene Versionen des Quellcodes
- zeigt wer wann was geändert hat
- ältere Versionen des Codes sind wiederherstellbar
- mehrere „Zweige“ eines Projekts können verwaltet werden
- kontrolliert den Zugriff der Entwickler auf den Quellcode

Versionshistorie

Versionshistorie ...

- wird bei der Softwareentwicklung verwendet
- verwaltet verschiedene Versionen des Quellcodes
- zeigt wer wann was geändert hat
- ältere Versionen des Codes sind wiederherstellbar
- mehrere „Zweige“ eines Projekts können verwaltet werden
- kontrolliert den Zugriff der Entwickler auf den Quellcode

Versionshistorie

Versionshistorie . . .

- wird bei der Softwareentwicklung verwendet
- verwaltet verschiedene Versionen des Quellcodes
- zeigt wer wann was geändert hat
- ältere Versionen des Codes sind wiederherstellbar
- mehrere „Zweige“ eines Projekts können verwaltet werden
- kontrolliert den Zugriff der Entwickler auf den Quellcode

Versionshistorie

Versionshistorie . . .

- wird bei der Softwareentwicklung verwendet
- verwaltet verschiedene Versionen des Quellcodes
- zeigt wer wann was geändert hat
- ältere Versionen des Codes sind wiederherstellbar
- mehrere „Zweige“ eines Projekts können verwaltet werden
- kontrolliert den Zugriff der Entwickler auf den Quellcode

Versionshistorie

Versionshistorie . . .

- wird bei der Softwareentwicklung verwendet
- verwaltet verschiedene Versionen des Quellcodes
- zeigt wer wann was geändert hat
- ältere Versionen des Codes sind wiederherstellbar
- mehrere „Zweige“ eines Projekts können verwaltet werden
- kontrolliert den Zugriff der Entwickler auf den Quellcode

Versionshistorie

Versionshistorie . . .

- wird bei der Softwareentwicklung verwendet
- verwaltet verschiedene Versionen des Quellcodes
- zeigt wer wann was geändert hat
- ältere Versionen des Codes sind wiederherstellbar
- mehrere „Zweige“ eines Projekts können verwaltet werden
- kontrolliert den Zugriff der Entwickler auf den Quellcode

Informationen aus der Versionshistorie

Informationen aus der Versionshistorie lassen sich nutzen, ...

- um weitere Änderungen vorzuschlagen
- um das Debugging zu erleichtern
- um problematische und fehleranfällige Module zu identifizieren
- zur Abschätzung ...
 - von Release-Intervallen
 - von Codewachstum
 - von Wartungskosten
 - der Fehlerzahl

Informationen aus der Versionshistorie

Informationen aus der Versionshistorie lassen sich nutzen, ...

- um weitere Änderungen vorzuschlagen
- um das Debugging zu erleichtern
- um problematische und fehleranfällige Module zu identifizieren
- zur Abschätzung ...
 - von Release-Intervallen
 - von Codewachstum
 - von Wartungskosten
 - der Fehlerzahl

Informationen aus der Versionshistorie

Informationen aus der Versionshistorie lassen sich nutzen, ...

- um weitere Änderungen vorzuschlagen
- um das Debugging zu erleichtern
- um problematische und fehleranfällige Module zu identifizieren
- zur Abschätzung ...
 - von Release-Intervallen
 - von Codewachstum
 - von Wartungskosten
 - der Fehlerzahl

Informationen aus der Versionshistorie

Informationen aus der Versionshistorie lassen sich nutzen, ...

- um weitere Änderungen vorzuschlagen
- um das Debugging zu erleichtern
- um problematische und fehleranfällige Module zu identifizieren
- zur Abschätzung ...
 - von Release-Intervallen
 - von Codewachstum
 - von Wartungskosten
 - der Fehlerzahl

Versions-Kontrollsysteme

Versions-Kontrollsysteme arbeiten . . .

- dateibasiert *oder*
- änderungsbasiert

Versions-Kontrollsysteme

Versions-Kontrollsysteme arbeiten ...

- dateibasiert *oder*
- änderungsbasiert

Beispiele

- RCS (Revision Control System)
- CVS (Concurrent Versions System)
- Subversion
- Microsoft Visual SourceSafe (VSS)
- Telelogic SYNERGY

Versions-Kontrollsysteme

Versions-Kontrollsysteme arbeiten ...

- dateibasiert *oder*
- änderungsbasiert

Beispiele

- RCS (Revision Control System)
- CVS (Concurrent Versions System)
- Subversion
- Microsoft Visual SourceSafe (VSS)
- Telelogic SYNERGY

Versions-Kontrollsysteme

Versions-Kontrollsysteme arbeiten ...

- dateibasiert *oder*
- änderungsbasiert

Beispiele

- RCS (Revision Control System)
- CVS (Concurrent Versions System)
- Subversion
- Microsoft Visual SourceSafe (VSS)
- Telelogic SYNERGY

Versions-Kontrollsysteme

Versions-Kontrollsysteme arbeiten ...

- dateibasiert *oder*
- änderungsbasiert

Beispiele

- RCS (Revision Control System)
- CVS (Concurrent Versions System)
- Subversion
- Microsoft Visual SourceSafe (VSS)
- Telelogic SYNERGY

Versions-Kontrollsysteme

Versions-Kontrollsysteme arbeiten ...

- dateibasiert *oder*
- änderungsbasiert

Beispiele

- RCS (Revision Control System)
- CVS (Concurrent Versions System)
- Subversion
- Microsoft Visual SourceSafe (VSS)
- Telelogic SYNERGY

Übersicht

- 1 Einführung
- 2 eROSE
 - Einführung
 - Technik
 - Auswertung
- 3 Weitere Ansätze
- 4 Zusammenfassung

Übersicht

- 1 Einführung
- 2 eROSE
 - Einführung
 - Technik
 - Auswertung
- 3 Weitere Ansätze
- 4 Zusammenfassung

eROSE: Einführung

- Abkürzung für „Reengineering of Software Evolution“
- am Lehrstuhl für Softwaretechnik der Universität des Saarlandes entwickelt
- Plugin für Eclipse
- zeigt mit Hilfe von Datamining auf der Versionshistorie verwandte Änderungen
- arbeitet ähnlich dem Vorschlagssystem von Amazon nach dem Prinzip “Programmers who changed these functions also changed ...”

eROSE: Einführung

- Abkürzung für „Reengineering of Software Evolution“
- am Lehrstuhl für Softwaretechnik der Universität des Saarlandes entwickelt
- Plugin für Eclipse
- zeigt mit Hilfe von Datamining auf der Versionshistorie verwandte Änderungen
- arbeitet ähnlich dem Vorschlagssystem von Amazon nach dem Prinzip “Programmers who changed these functions also changed ...”

eROSE: Einführung

- Abkürzung für „Reengineering of Software Evolution“
- am Lehrstuhl für Softwaretechnik der Universität des Saarlandes entwickelt
- Plugin für Eclipse
- zeigt mit Hilfe von Datamining auf der Versionshistorie verwandte Änderungen
- arbeitet ähnlich dem Vorschlagssystem von Amazon nach dem Prinzip “Programmers who changed these functions also changed ...”

eROSE: Einführung

- Abkürzung für „Reengineering of Software Evolution“
- am Lehrstuhl für Softwaretechnik der Universität des Saarlandes entwickelt
- Plugin für Eclipse
- zeigt mit Hilfe von Datamining auf der Versionshistorie verwandte Änderungen
- arbeitet ähnlich dem Vorschlagssystem von Amazon nach dem Prinzip “Programmers who changed these functions also changed ...”

eROSE: Einführung

- Abkürzung für „Reengineering of Software Evolution“
- am Lehrstuhl für Softwaretechnik der Universität des Saarlandes entwickelt
- Plugin für Eclipse
- zeigt mit Hilfe von Datamining auf der Versionshistorie verwandte Änderungen
- arbeitet ähnlich dem Vorschlagssystem von Amazon nach dem Prinzip “Programmers who changed these functions also changed ...”

Übersicht

- 1 Einführung
- 2 eROSE
 - Einführung
 - Technik
 - Auswertung
- 3 Weitere Ansätze
- 4 Zusammenfassung

eROSE: Begriffe

Transaktion:

- besteht aus Änderungen an mehreren Stellen im Code
- aus mehreren Commits des gleichen Entwicklers innerhalb einer Zeitspanne von mehreren Minuten gruppiert

Entität:

- syntaktische Komponente
 - Quellcode, Konfiguration, Buildskripte, ...
 - Quellcode als Typ
 - Quellcode-Komponenten, ...

eROSE: Begriffe

Transaktion:

- besteht aus Änderungen an mehreren Stellen im Code
- aus mehreren Commits des gleichen Entwicklers innerhalb einer Zeitspanne von mehreren Minuten gruppiert

Entität:

- syntaktische Komponente
- z.B. Konstante, Array, Methode, Klasse, Datei
- dargestellt als Tripel:
(Dateiname, Komponentenart, Bezeichner)

eROSE: Begriffe

Transaktion:

- besteht aus Änderungen an mehreren Stellen im Code
- aus mehreren Commits des gleichen Entwicklers innerhalb einer Zeitspanne von mehreren Minuten gruppiert

Entität:

- syntaktische Komponente
- z.B. Konstante, Array, Methode, Klasse, Datei
- dargestellt als Tripel:
(Dateiname, Komponentenart, Bezeichner)

eROSE: Begriffe

Transaktion:

- besteht aus Änderungen an mehreren Stellen im Code
- aus mehreren Commits des gleichen Entwicklers innerhalb einer Zeitspanne von mehreren Minuten gruppiert

Entität:

- syntaktische Komponente
- z.B. Konstante, Array, Methode, Klasse, Datei
- dargestellt als Tripel:
(Dateiname, Komponentenart, Bezeichner)

eROSE: Begriffe

Transaktion:

- besteht aus Änderungen an mehreren Stellen im Code
- aus mehreren Commits des gleichen Entwicklers innerhalb einer Zeitspanne von mehreren Minuten gruppiert

Entität:

- syntaktische Komponente
- z.B. Konstante, Array, Methode, Klasse, Datei
- dargestellt als Tripel:
(Dateiname, Komponentenart, Bezeichner)

eROSE: Entitäten und Regeln

Beispiel (Entitätenmenge)

$$\left\{ \begin{array}{l} (\text{CompPrefPage.java}, \textit{method}, \text{initDefaults}()), \\ (\text{CompPrefPage.java}, \textit{field}, \text{fKeys}[]), \\ (\text{CompPrefPage.java}, \textit{class}, \text{CompPrefPage}), \\ (\text{CompPrefPage.java}, \textit{file}, \text{CompPrefPage.java}), \\ \dots \end{array} \right\}$$

eROSE: Entitäten und Regeln

Beispiel (Entitätenmenge)

$$\left\{ \begin{array}{l} (\text{CompPrefPage.java}, \textit{method}, \text{initDefaults}()), \\ (\text{CompPrefPage.java}, \textit{field}, \text{fKeys} []), \\ (\text{CompPrefPage.java}, \textit{class}, \text{CompPrefPage}), \\ (\text{CompPrefPage.java}, \textit{file}, \text{CompPrefPage.java}), \\ \dots \end{array} \right\}$$

Beispiel (Regel)

$$\{(\text{CompPrefPage.java}, \textit{field}, \text{fKeys} [])\}$$

$$\Rightarrow \left\{ \begin{array}{l} (\text{CompPrefPage.java}, \textit{method}, \text{initDefaults}()), \\ (\text{plugin.properties}, \textit{file}, \text{plugin.properties}) \end{array} \right\}$$

eROSE: Unterstützung / Konfidenz von Regeln

Unterstützung:

- gibt an, aus wie vielen Transaktionen die Regel abgeleitet wurde

Konfidenz:

- gibt die Stärke des Aktionsteils der Regel an
- wird berechnet als:

$$\frac{\text{Unterstützung}}{\text{Anzahl der Transaktionen, die den Bedingungsteil betreffen}}$$

eROSE: Unterstützung / Konfidenz von Regeln

Unterstützung:

- gibt an, aus wie vielen Transaktionen die Regel abgeleitet wurde

Konfidenz:

- gibt die Stärke des Aktionsteils der Regel an
- wird berechnet als:

$$\frac{\text{Unterstützung}}{\text{Anzahl der Transaktionen, die den Bedingungsteil betreffen}}$$

eROSE: Unterstützung / Konfidenz von Regeln

Unterstützung:

- gibt an, aus wie vielen Transaktionen die Regel abgeleitet wurde

Konfidenz:

- gibt die Stärke des Aktionsteils der Regel an
- wird berechnet als:

$$\frac{\text{Unterstützung}}{\text{Anzahl der Transaktionen, die den Bedingungsteil betreffen}}$$

Übersicht

- 1 Einführung
- 2 eROSE
 - Einführung
 - Technik
 - **Auswertung**
- 3 Weitere Ansätze
- 4 Zusammenfassung

eROSE: Auswertung

Navigation durch den Code:

- nach einer geänderten Entität kann eROSE 15 Prozent aller weiteren zu ändernden Entitäten der gleichen Transaktion vorschlagen
- GCC: 28 Prozent / KOffice: 8 Prozent
- in 64 Prozent aller Transaktionen verweisen die ersten drei Vorschläge auf die richtigen Stellen im Code

Verhinderung von Fehlern:

- wenn in einer Transaktion eine Änderung vergessen wurde, kann eROSE im Schnitt nur vier Prozent dieser Fälle erkennen

eROSE: Auswertung

Navigation durch den Code:

- nach einer geänderten Entität kann eROSE 15 Prozent aller weiteren zu ändernden Entitäten der gleichen Transaktion vorschlagen
- GCC: 28 Prozent / KOffice: 8 Prozent
- in 64 Prozent aller Transaktionen verweisen die ersten drei Vorschläge auf die richtigen Stellen im Code

Verhinderung von Fehlern:

- wenn in einer Transaktion eine Änderung vergessen wurde, kann eROSE im Schnitt nur vier Prozent dieser Fälle erkennen

eROSE: Auswertung

Navigation durch den Code:

- nach einer geänderten Entität kann eROSE 15 Prozent aller weiteren zu ändernden Entitäten der gleichen Transaktion vorschlagen
- GCC: 28 Prozent / KOffice: 8 Prozent
- in 64 Prozent aller Transaktionen verweisen die ersten drei Vorschläge auf die richtigen Stellen im Code

Verhinderung von Fehlern:

- wenn in einer Transaktion eine Änderung vergessen wurde, kann eROSE im Schnitt nur vier Prozent dieser Fälle erkennen
- durchschnittlich ist jeder zweite Vorschlag korrekt

eROSE: Auswertung

Navigation durch den Code:

- nach einer geänderten Entität kann eROSE 15 Prozent aller weiteren zu ändernden Entitäten der gleichen Transaktion vorschlagen
- GCC: 28 Prozent / KOffice: 8 Prozent
- in 64 Prozent aller Transaktionen verweisen die ersten drei Vorschläge auf die richtigen Stellen im Code

Verhinderung von Fehlern:

- wenn in einer Transaktion eine Änderung vergessen wurde, kann eROSE im Schnitt nur vier Prozent dieser Fälle erkennen
- durchschnittlich ist jeder zweite Vorschlag korrekt

eROSE: Auswertung

Navigation durch den Code:

- nach einer geänderten Entität kann eROSE 15 Prozent aller weiteren zu ändernden Entitäten der gleichen Transaktion vorschlagen
- GCC: 28 Prozent / KOffice: 8 Prozent
- in 64 Prozent aller Transaktionen verweisen die ersten drei Vorschläge auf die richtigen Stellen im Code

Verhinderung von Fehlern:

- wenn in einer Transaktion eine Änderung vergessen wurde, kann eROSE im Schnitt nur vier Prozent dieser Fälle erkennen
- durchschnittlich ist jeder zweite Vorschlag korrekt

eROSE: Auswertung

Abgeschlossenheit:

- in nur zwei Prozent aller Fälle schlägt eROSE noch weitere Änderungen vor, obwohl bereits alle notwendigen Änderungen vorgenommen wurden

Granularität:

- mit Vorschlägen nur auf Dateiebene kann eROSE 26 statt 15 Prozent der weiteren Änderungen vorschlagen
- die Korrektheit der ersten drei Vorschläge steigt von 64 auf 70 Prozent

eROSE: Auswertung

Abgeschlossenheit:

- in nur zwei Prozent aller Fälle schlägt eROSE noch weitere Änderungen vor, obwohl bereits alle notwendigen Änderungen vorgenommen wurden

Granularität:

- mit Vorschlägen nur auf Dateiebene kann eROSE 26 statt 15 Prozent der weiteren Änderungen vorschlagen
- die Korrektheit der ersten drei Vorschläge steigt von 64 auf 70 Prozent

eROSE: Auswertung

Abgeschlossenheit:

- in nur zwei Prozent aller Fälle schlägt eROSE noch weitere Änderungen vor, obwohl bereits alle notwendigen Änderungen vorgenommen wurden

Granularität:

- mit Vorschlägen nur auf Dateiebene kann eROSE 26 statt 15 Prozent der weiteren Änderungen vorschlagen
- die Korrektheit der ersten drei Vorschläge steigt von 64 auf 70 Prozent

Übersicht

- 1 Einführung
- 2 eROSE
- 3 Weitere Ansätze**
 - Hipikat
 - CVSSearch
 - Version Editor
 - VssConneXion
- 4 Zusammenfassung

Übersicht

- 1 Einführung
- 2 eROSE
- 3 Weitere Ansätze**
 - **Hipikat**
 - CVSSearch
 - Version Editor
 - VssConneXion
- 4 Zusammenfassung

Hipikat: Vorstellung

- gemeinsames Projekt der University of British Columbia, des IBM Ottawa Software Lab und des National Research Council of Canada
- Plugin für Eclipse
- berücksichtigt nicht nur Versionshistorien, sondern auch ...
 - die gesamte Entwicklungssysteme wie CVS, Subversion, CVS, ClearCase, etc.
 - die Teamorganisation
 - die Meetings und Newsgroups
- bietet Vorschläge nur auf Dateiebene

Hipikat: Vorstellung

- gemeinsames Projekt der University of British Columbia, des IBM Ottawa Software Lab und des National Research Council of Canada
- Plugin für Eclipse
- berücksichtigt nicht nur Versionshistorien, sondern auch ...
 - ✦ Daten aus Bugtracking-Systemen wie Bugzilla,
 - ✦ Online-Dokumentationen
 - ✦ Mailinglisten und Newsgroups
- bietet Vorschläge nur auf Dateiebene

Hipikat: Vorstellung

- gemeinsames Projekt der University of British Columbia, des IBM Ottawa Software Lab und des National Research Council of Canada
- Plugin für Eclipse
- berücksichtigt nicht nur Versionshistorien, sondern auch ...
 - Daten aus Bugtracking-Systemen wie Bugzilla,
 - Online-Dokumentationen
 - Mailinglisten und Newsgroups
- bietet Vorschläge nur auf Dateiebene

Hipikat: Vorstellung

- gemeinsames Projekt der University of British Columbia, des IBM Ottawa Software Lab und des National Research Council of Canada
- Plugin für Eclipse
- berücksichtigt nicht nur Versionshistorien, sondern auch ...
 - Daten aus Bugtracking-Systemen wie Bugzilla,
 - Online-Dokumentationen
 - Mailinglisten und Newsgroups
- bietet Vorschläge nur auf Dateiebene

Hipikat: Vorstellung

- gemeinsames Projekt der University of British Columbia, des IBM Ottawa Software Lab und des National Research Council of Canada
- Plugin für Eclipse
- berücksichtigt nicht nur Versionshistorien, sondern auch ...
 - Daten aus Bugtracking-Systemen wie Bugzilla,
 - Online-Dokumentationen
 - Mailinglisten und Newsgroups
- bietet Vorschläge nur auf Dateiebene

Hipikat: Vorstellung

- gemeinsames Projekt der University of British Columbia, des IBM Ottawa Software Lab und des National Research Council of Canada
- Plugin für Eclipse
- berücksichtigt nicht nur Versionshistorien, sondern auch ...
 - Daten aus Bugtracking-Systemen wie Bugzilla,
 - Online-Dokumentationen
 - Mailinglisten und Newsgroups
- bietet Vorschläge nur auf Dateiebene

Hipikat: Vorstellung

- gemeinsames Projekt der University of British Columbia, des IBM Ottawa Software Lab und des National Research Council of Canada
- Plugin für Eclipse
- berücksichtigt nicht nur Versionshistorien, sondern auch ...
 - Daten aus Bugtracking-Systemen wie Bugzilla,
 - Online-Dokumentationen
 - Mailinglisten und Newsgroups
- bietet Vorschläge nur auf Dateiebene

Hipikat: Beispiel

Search [X]

File Search Bugzilla Search Hipikat Search Java Search

Bug id or summary search terms
 all words

Comment contains: all words

Product	Component	Version	Milestone
AJDT	Access Control	0.5	2.0 M1
AspectJ	Ant	1.0	Alpha1
CDT	Commandline	2.0	Alpha2

Status	Resolution	Severity	Priority	Hardware	OS
Unconfirmed	Fixed	blocker	P1	All	All
New	Invalid	critical	P2	Macintosh	AIK Motif
Assigned	Won'tfix	major	P3	PC	Windows 95

bug owner reporter CC list commenter

Email contains: substring

Customize... Search Cancel

Beispiel

especially over slow connection. Make it user pref?

Hipikat (Bug 116)

Type	Name	Reason	Confidence
bugzilla	Bug 80 - Should be able to use delete key in repo view (1GFXP...	Text similarity	0.3728732...
bugzilla	Bug 2219 - Refresh from local should be renamed (1GEAOT8)	Text similarity	0.3180000...
bugzilla	Bug 59 - new repo connection should offer to open repos vie...	Text similarity	0.2868199...
bugzilla	Bug 11419 - Automatically add version to repo view when tag...	Text similarity	0.2752248...
bugzilla	Bug 133 - Show in resource history while already in resource h...	Text similarity	0.2655328...
bugzilla	Bug 2305 - [JFace Text] Text selection off by one when doubl...	Text similarity	0.2602417...
bugzilla	Bug 18075 - adding a breakpoint refreshes outline view...	Text similarity	0.2500408...

Tasks | Hipikat Results | Search

Übersicht

- 1 Einführung
- 2 eROSE
- 3 Weitere Ansätze**
 - Hipikat
 - **CVSSearch**
 - Version Editor
 - VssConneXion
- 4 Zusammenfassung

CVSSearch: Vorstellung

- an der University of New South Wales entwickelt
- sucht nach Teilen des Quellcodes anhand von CVS-Kommentaren
- Vorteil gegenüber Kommentaren im Code:
 - bei größeren Open-Source-Projekten sind kaum Kommentare im Code vorhanden oder nur von schlechter Qualität
 - CVS-Kommentare dienen hauptsächlich dazu, die anderen Entwickler über die aktuellen Änderungen zu informieren
- kombiniert die Ergebnisse aus der Versionshistorie mit den Ergebnissen von *grep*
- somit Suche aus zwei Perspektiven:
 - Blick in den Code
 - Blick auf das, was die Entwickler über den Code schreiben

CVSSearch: Vorstellung

- an der University of New South Wales entwickelt
- sucht nach Teilen des Quellcodes anhand von CVS-Kommentaren
- Vorteil gegenüber Kommentaren im Code:
 - bei größeren Open-Source-Projekten sind kaum Kommentare im Code vorhanden oder nur von schlechter Qualität
 - CVS-Kommentare dienen hauptsächlich dazu, die anderen Entwickler über die aktuellen Änderungen zu informieren
- kombiniert die Ergebnisse aus der Versionshistorie mit den Ergebnissen von *grep*
- somit Suche aus zwei Perspektiven:
 - Blick in den Code
 - Blick auf das, was die Entwickler über den Code schreiben

CVSSearch: Vorstellung

- an der University of New South Wales entwickelt
- sucht nach Teilen des Quellcodes anhand von CVS-Kommentaren
- Vorteil gegenüber Kommentaren im Code:
 - bei größeren Open-Source-Projekten sind kaum Kommentare im Code vorhanden oder nur von schlechter Qualität
 - CVS-Kommentare dienen hauptsächlich dazu, die anderen Entwickler über die aktuellen Änderungen zu informieren
- kombiniert die Ergebnisse aus der Versionshistorie mit den Ergebnissen von *grep*
- somit Suche aus zwei Perspektiven:
 - Blick in den Code
 - Blick auf das, was die Entwickler über den Code schreiben

CVSSearch: Vorstellung

- an der University of New South Wales entwickelt
- sucht nach Teilen des Quellcodes anhand von CVS-Kommentaren
- Vorteil gegenüber Kommentaren im Code:
 - bei größeren Open-Source-Projekten sind kaum Kommentare im Code vorhanden oder nur von schlechter Qualität
 - CVS-Kommentare dienen hauptsächlich dazu, die anderen Entwickler über die aktuellen Änderungen zu informieren
- kombiniert die Ergebnisse aus der Versionshistorie mit den Ergebnissen von *grep*
- somit Suche aus zwei Perspektiven:
 - Blick in den Code
 - Blick auf das, was die Entwickler über den Code schreiben

CVSSearch: Vorstellung

- an der University of New South Wales entwickelt
- sucht nach Teilen des Quellcodes anhand von CVS-Kommentaren
- Vorteil gegenüber Kommentaren im Code:
 - bei größeren Open-Source-Projekten sind kaum Kommentare im Code vorhanden oder nur von schlechter Qualität
 - CVS-Kommentare dienen hauptsächlich dazu, die anderen Entwickler über die aktuellen Änderungen zu informieren
- kombiniert die Ergebnisse aus der Versionshistorie mit den Ergebnissen von *grep*
- somit Suche aus zwei Perspektiven:
 - Blick in den Code
 - Blick auf das, was die Entwickler über den Code schreiben

CVSSearch: Beispiel



Enter keyword(s) to search for:

Select Repository:

[Browse Files/Commits](#)

[Browse Library Class/Function Usage Patterns](#)

Search for:

Tips

- Use `in:` at the end of keywords to select package to search in. For example, `drag drop in:kdebase/konqueror;kdepin/korganizer` searches for menus under `kdebase/konqueror` and `kdepin/korganizer`; default searches for keywords under all packages. It does not have to be full path, for example, you can use `in:keyword` instead of `in:koffice/keyword`, however you cannot use `in:kw`.
- Keywords are not case-sensitive and stemmed. (e.g. searching for 'fishes' will match 'FISH', 'fishes', 'fishing'...)
- Commit comments are useful not only for finding changes made to the code (e.g. fixed footnotes) but also identifying where certain functionality is implemented (e.g., the code changed in fixing footnotes reveals the regions of code that implement footnotes).
- Searches across all applications can be useful in finding out how a certain task is accomplished using library functions.

Please send all bugs/comments to cvssearch@cse.msuw.edu

The project page is <http://cvssearch.sourceforge.net>.

CVSSearch is under the GPL.

Übersicht

- 1 Einführung
- 2 eROSE
- 3 Weitere Ansätze**
 - Hipikat
 - CVSSearch
 - Version Editor**
 - VssConneXion
- 4 Zusammenfassung

Version Editor: Vorstellung

- von David L. Atkins an den Bell Laboratories entwickelt
- setzt auf die Editoren *vi* und *Emacs* auf
- zeigt nicht nur den aktuellen Code
- sondern auch diejenigen Zeilen, die seit dem letzten Checkout geändert oder gelöscht wurden
- erleichtert dadurch die Suche nach Fehlern
- zusätzlich werden für die aktuelle Zeile Informationen zur letzten Änderung angezeigt

Version Editor: Vorstellung

- von David L. Atkins an den Bell Laboratories entwickelt
- setzt auf die Editoren *vi* und *Emacs* auf
- zeigt nicht nur den aktuellen Code
- sondern auch diejenigen Zeilen, die seit dem letzten Checkout geändert oder gelöscht wurden
- erleichtert dadurch die Suche nach Fehlern
- zusätzlich werden für die aktuelle Zeile Informationen zur letzten Änderung angezeigt

Version Editor: Vorstellung

- von David L. Atkins an den Bell Laboratories entwickelt
- setzt auf die Editoren *vi* und *Emacs* auf
- zeigt nicht nur den aktuellen Code
- sondern auch diejenigen Zeilen, die seit dem letzten Checkout geändert oder gelöscht wurden
- erleichtert dadurch die Suche nach Fehlern
- zusätzlich werden für die aktuelle Zeile Informationen zur letzten Änderung angezeigt

Version Editor: Vorstellung

- von David L. Atkins an den Bell Laboratories entwickelt
- setzt auf die Editoren *vi* und *Emacs* auf
- zeigt nicht nur den aktuellen Code
- sondern auch diejenigen Zeilen, die seit dem letzten Checkout geändert oder gelöscht wurden
- erleichtert dadurch die Suche nach Fehlern
- zusätzlich werden für die aktuelle Zeile Informationen zur letzten Änderung angezeigt

Version Editor: Vorstellung

- von David L. Atkins an den Bell Laboratories entwickelt
- setzt auf die Editoren *vi* und *Emacs* auf
- zeigt nicht nur den aktuellen Code
- sondern auch diejenigen Zeilen, die seit dem letzten Checkout geändert oder gelöscht wurden
- erleichtert dadurch die Suche nach Fehlern
- zusätzlich werden für die aktuelle Zeile Informationen zur letzten Änderung angezeigt

Version Editor: Vorstellung

- von David L. Atkins an den Bell Laboratories entwickelt
- setzt auf die Editoren *vi* und *Emacs* auf
- zeigt nicht nur den aktuellen Code
- sondern auch diejenigen Zeilen, die seit dem letzten Checkout geändert oder gelöscht wurden
- erleichtert dadurch die Suche nach Fehlern
- zusätzlich werden für die aktuelle Zeile Informationen zur letzten Änderung angezeigt

Version Editor: Beispiel

```
String FindSource(String base, String dir) {
    DIR * dirp = opendir(dir);
    String result;    // The filename, if found
    for (int i = 0; i < NS; ++i) {        // Loop over suffix list
        String tmp = base + suffix[i];    // Target name to find
        for (dirent *de = readdir(dirp); de != NULL; de = readdir(dirp))
            if (tmp == de->d_name) {      // We found it, stop looking
                result = tmp;
                break;
                return tmp;
            }
        rewinddir(dirp);
    }
    closedir(dirp);
    return result; // Return the found name (may be null)
    return "";    // No match was found
}
```

Deleted by MR 595 by vz,97/11/15, approved [Stop source search at 1st match]
MR 467 by dla,97/09/21,integrated [Find source using list of suffixes]
"findsource.c", line 15 of 23

Übersicht

- 1 Einführung
- 2 eROSE
- 3 Weitere Ansätze**
 - Hipikat
 - CVSSearch
 - Version Editor
 - VssConneXion**
- 4 Zusammenfassung

VssConneXion: Vorstellung

- von EPocalipse Software entwickelt
- Plugin für Borland Delphi
- bietet Anbindung von Delphi an Microsoft Visual SourceSafe
- „Source Analysis“ mit vier Ansichten:
 - Änderungen der einzelnen Zeilen mit Angabe des Änderungsdatums
 - Änderungen des Entwicklers, der sich an der Visual SourceSafe-Datenbank angemeldet hat
 - Entwickler, die die einzelnen Zeilen geänderten Zeilen geändert haben
 - Anzahl der Änderungen der einzelnen Zeilen

VssConneXion: Vorstellung

- von EPocalipse Software entwickelt
- Plugin für Borland Delphi
- bietet Anbindung von Delphi an Microsoft Visual SourceSafe
- „Source Analysis“ mit vier Ansichten:
 - Änderungen der einzelnen Zeilen mit Angabe des Änderungsdatums
 - Änderungen des Entwicklers, der sich an der Visual SourceSafe-Datenbank angemeldet hat
 - Entwickler, die die einzelnen Zeilen geänderten Zeilen geändert haben
 - Anzahl der Änderungen der einzelnen Zeilen

VssConneXion: Vorstellung

- von EPocalipse Software entwickelt
- Plugin für Borland Delphi
- bietet Anbindung von Delphi an Microsoft Visual SourceSafe
- „Source Analysis“ mit vier Ansichten:
 - Änderungen der einzelnen Zeilen mit Angabe des Änderungsdatums
 - Änderungen des Entwicklers, der sich an der Visual SourceSafe-Datenbank angemeldet hat
 - Entwickler, die die einzelnen Zeilen geänderten Zeilen geändert haben
 - Anzahl der Änderungen der einzelnen Zeilen

VssConneXion: Vorstellung

- von EPocalypse Software entwickelt
- Plugin für Borland Delphi
- bietet Anbindung von Delphi an Microsoft Visual SourceSafe
- „Source Analysis“ mit vier Ansichten:
 - Änderungen der einzelnen Zeilen mit Angabe des Änderungsdatums
 - Änderungen des Entwicklers, der sich an der Visual SourceSafe-Datenbank angemeldet hat
 - Entwickler, die die einzelnen Zeilen geänderten Zeilen geändert haben
 - Anzahl der Änderungen der einzelnen Zeilen

VssConneXion: Beispiel

The screenshot shows the Source Analysis tool interface. The main window displays a code editor with the following code:

```

13 function
14 var
15   Len: DWORD;
16 begin
17   Len:=0;
18   GetUserName(nil, Len);
19   if Len>0 then
20   begin
21     SetLength(Result, Len);
22     GetUserName(PChar(Result), Len);
23     SetLength(Result, StrLen(PChar(Result)));
24   end
25   else
26     Result:='';
27 end;
28
29 procedure SetEnv(const Name, Value: string);
30 begin
31   SetEnvironmentVariable(PChar(Name), PChar(Value));
32 end;
33
34 end.

```

A dropdown menu is open over the code, listing the following options:

- Number of Changes
- Changes by date
- My Changes by date
- Changes by User
- Number of Changes

The Line History window on the right shows the following entry:

#	User	Date
3	John	10/9/2003 12:2...

Below the table, the text "Added a new SetEnv function" is displayed.

At the bottom of the tool, there are two buttons: "1 Change" and "2 Changes".

Übersicht

- 1 Einführung
- 2 eROSE
- 3 Weitere Ansätze
- 4 Zusammenfassung**

Zusammenfassung

eROSE und Hipikat:

- zeigen Vorschläge für weitere Änderungen an
- benutzen Datamining, um Regeln oder Verknüpfungen zu berechnen

CVSSearch:

- ermöglicht Suche nach Code-Fragmenten
- Suchen aus zwei Perspektiven

Version Editor und VssConneXion:

- erleichtern Debugging

Zusammenfassung

eROSE und Hipikat:

- zeigen Vorschläge für weitere Änderungen an
- benutzen Datamining, um Regeln oder Verknüpfungen zu berechnen

CVSSearch:

- ermöglicht Suche nach Code-Fragmenten
- sucht aus zwei Perspektiven

Version Editor und VssConneXion:

- erleichtern Debugging

Zusammenfassung

eROSE und Hipikat:

- zeigen Vorschläge für weitere Änderungen an
- benutzen Datamining, um Regeln oder Verknüpfungen zu berechnen

CVSSearch:

- ermöglicht Suche nach Code-Fragmenten
- sucht aus zwei Perspektiven

Version Editor und VssConneXion:

- erleichtern Debugging

Zusammenfassung

eROSE und Hipikat:

- zeigen Vorschläge für weitere Änderungen an
- benutzen Datamining, um Regeln oder Verknüpfungen zu berechnen

CVSSearch:

- ermöglicht Suche nach Code-Fragmenten
- sucht aus zwei Perspektiven

Version Editor und VssConneXion:

- erleichtern Debugging
- Version Editor zeigt geänderte und gelöschte Zeilen an
- VssConneXion zeigt Anzahl, Datum und Autoren der Änderungen an

Zusammenfassung

eROSE und Hipikat:

- zeigen Vorschläge für weitere Änderungen an
- benutzen Datamining, um Regeln oder Verknüpfungen zu berechnen

CVSSearch:

- ermöglicht Suche nach Code-Fragmenten
- sucht aus zwei Perspektiven

Version Editor und VssConneXion:

- erleichtern Debugging
- Version Editor zeigt geänderte und gelöschte Zeilen an
- VssConneXion zeigt Anzahl, Datum und Autoren der Änderungen an

Zusammenfassung

eROSE und Hipikat:

- zeigen Vorschläge für weitere Änderungen an
- benutzen Datamining, um Regeln oder Verknüpfungen zu berechnen

CVSSearch:

- ermöglicht Suche nach Code-Fragmenten
- sucht aus zwei Perspektiven

Version Editor und VssConneXion:

- erleichtern Debugging
- Version Editor zeigt geänderte und gelöschte Zeilen an
- VssConneXion zeigt Anzahl, Datum und Autoren der Änderungen an

Zusammenfassung

eROSE und Hipikat:

- zeigen Vorschläge für weitere Änderungen an
- benutzen Datamining, um Regeln oder Verknüpfungen zu berechnen

CVSSearch:

- ermöglicht Suche nach Code-Fragmenten
- sucht aus zwei Perspektiven

Version Editor und VssConneXion:

- erleichtern Debugging
- Version Editor zeigt geänderte und gelöschte Zeilen an
- VssConneXion zeigt Anzahl, Datum und Autoren der Änderungen an

Ausblick

eROSE:

- als Alpha-Version 0.0.5 zum Download verfügbar
- Version 0.1.0 ist für Juni 2006 geplant

Hipikat:

- Version 1.8.0 von Hipikat wurde im Juli 2003 fertiggestellt
- zur Zeit kein Download möglich

CVSSearch:

- Version 2.0 Beta2 war verfügbar
- zur Zeit nur direkter Zugriff auf den CVS-Server möglich

Version Editor:

- nicht zum Download verfügbar

VssConneXion:

- Version 4 als 30-Tage-Testversion und als Vollversion verfügbar

Ausblick

eROSE:

- als Alpha-Version 0.0.5 zum Download verfügbar
- Version 0.1.0 ist für Juni 2006 geplant

Hipikat:

- Version 1.8.0 von Hipikat wurde im Juli 2003 fertiggestellt
- zur Zeit kein Download möglich

CVSSearch:

- Version 2.0 Beta2 war verfügbar
- zur Zeit nur direkter Zugriff auf den CVS-Server möglich

Version Editor:

- nicht zum Download verfügbar

VssConneXion:

- Version 4 als 30-Tage-Testversion und als Vollversion verfügbar

Ausblick

eROSE:

- als Alpha-Version 0.0.5 zum Download verfügbar
- Version 0.1.0 ist für Juni 2006 geplant

Hipikat:

- Version 1.8.0 von Hipikat wurde im Juli 2003 fertiggestellt
- zur Zeit kein Download möglich

CVSSearch:

- Version 2.0 Beta2 war verfügbar
- zur Zeit nur direkter Zugriff auf den CVS-Server möglich

Version Editor:

- nicht zum Download verfügbar

VssConneXion:

- Version 4 als 30-Tage-Testversion und als Vollversion verfügbar

Ausblick

eROSE:

- als Alpha-Version 0.0.5 zum Download verfügbar
- Version 0.1.0 ist für Juni 2006 geplant

Hipikat:

- Version 1.8.0 von Hipikat wurde im Juli 2003 fertiggestellt
- zur Zeit kein Download möglich

CVSSearch:

- Version 2.0 Beta2 war verfügbar
- zur Zeit nur direkter Zugriff auf den CVS-Server möglich

Version Editor:

- nicht zum Download verfügbar

VssConneXion:

- Version 4 als 30-Tage-Testversion und als Vollversion verfügbar

Ausblick

eROSE:

- als Alpha-Version 0.0.5 zum Download verfügbar
- Version 0.1.0 ist für Juni 2006 geplant

Hipikat:

- Version 1.8.0 von Hipikat wurde im Juli 2003 fertiggestellt
- zur Zeit kein Download möglich

CVSSearch:

- Version 2.0 Beta2 war verfügbar
- zur Zeit nur direkter Zugriff auf den CVS-Server möglich

Version Editor:

- nicht zum Download verfügbar

VssConneXion:

- Version 4 als 30-Tage-Testversion und als Vollversion verfügbar