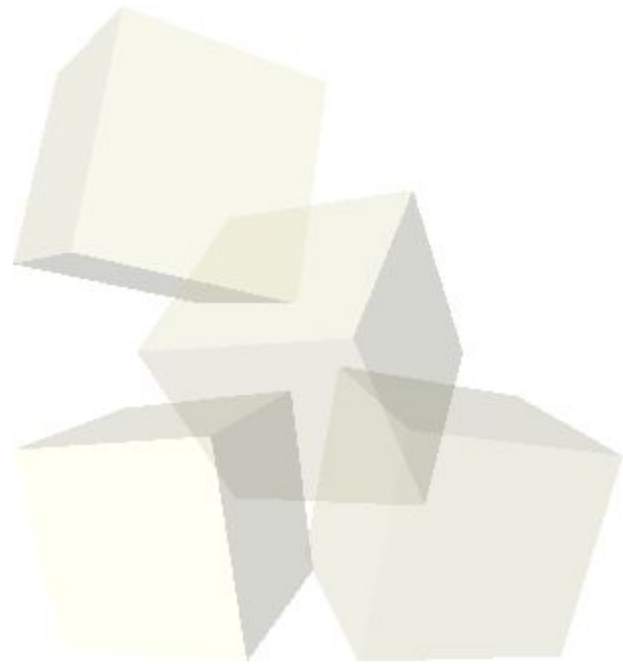




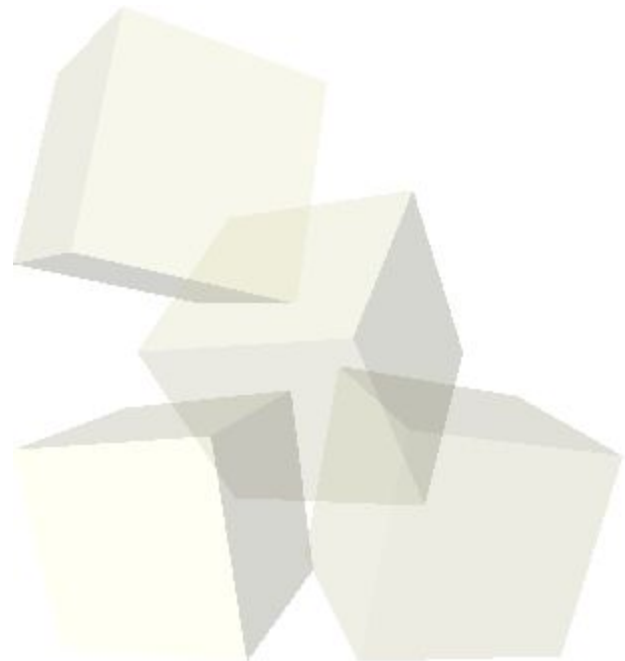
Projektorganisation





“Brooks' Law”

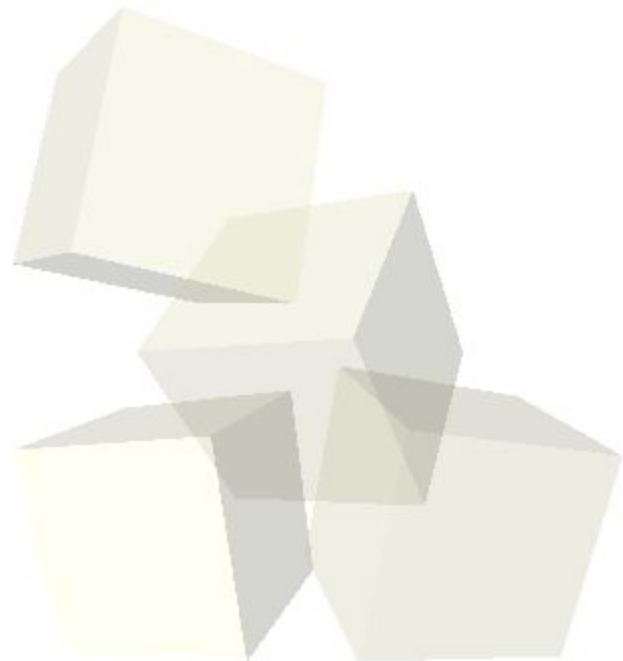
- Aus Fred Brooks - “The Mythical Man-Month”





“Brooks' Law”

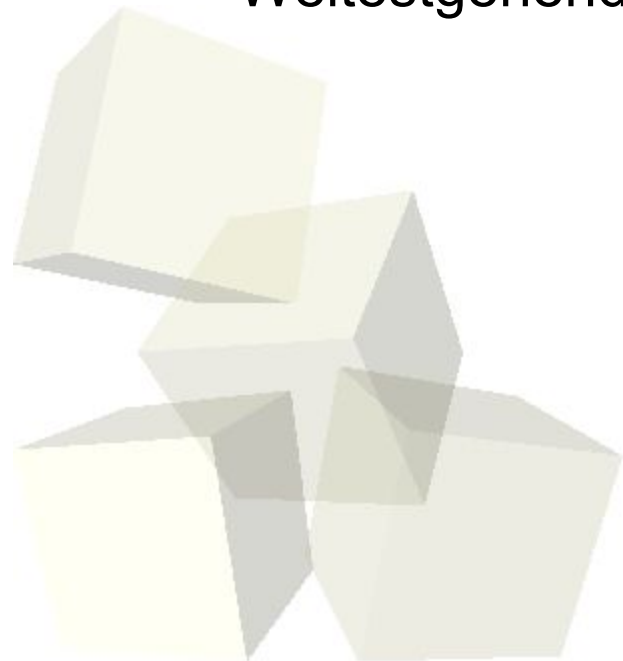
- Aus Fred Brooks - “The Mythical Man-Month”
- Mit der Anzahl der Programmierer wächst der Koordinationsaufwand wesentlich schneller als der tatsächliche Nutzen
 - nur kleine, überschaubare Teams sind effizient





“Brooks' Law”

- Aus Fred Brooks - “The Mythical Man-Month”
- Mit der Anzahl der Programmierer wächst der Koordinationsaufwand wesentlich schneller als der tatsächliche Nutzen
 - nur kleine, überschaubare Teams sind effizient
- Weitestgehend in der Softwareindustrie gültig und akzeptiert



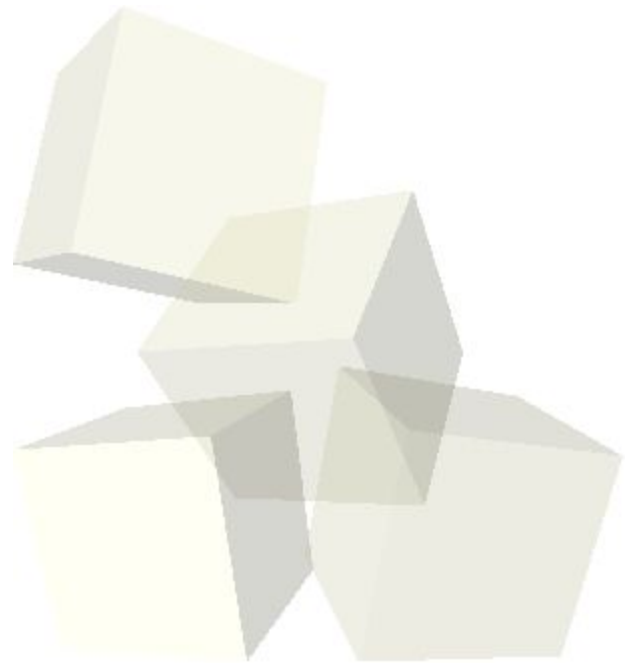


“Brooks' Law”

- Aus Fred Brooks - “The Mythical Man-Month”
- Mit der Anzahl der Programmierer wächst der Koordinationsaufwand wesentlich schneller als der tatsächliche Nutzen
 - nur kleine, überschaubare Teams sind effizient
- Weitestgehend in der Softwareindustrie gültig und akzeptiert
- Im Open Source Bereich scheinbar nicht mehr zutreffend
 - Selbst große Projekte wie der Linux Kernel scheinen es zu schaffen, den Koordinationsaufwand zu bewältigen

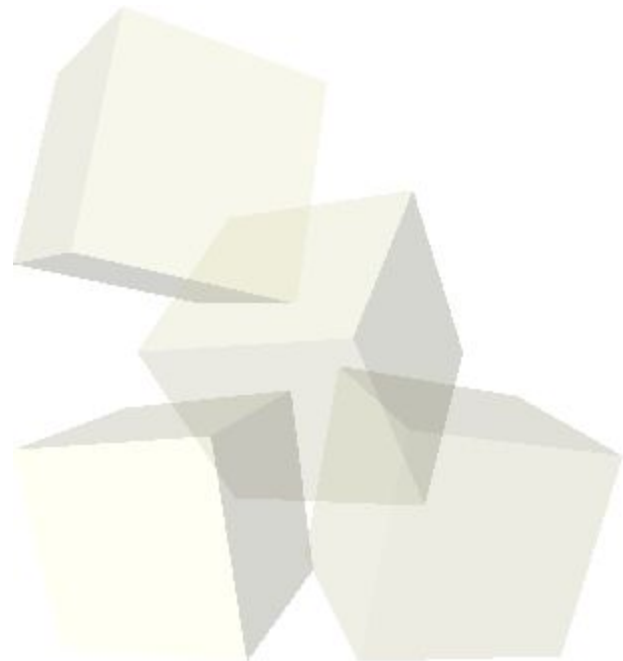


1. Wie entsteht ein OSS Projekt?



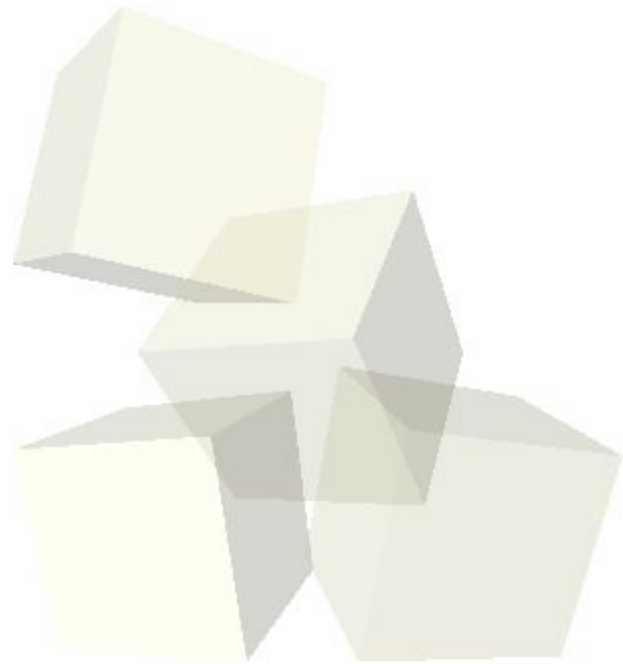


1. Wie entsteht ein OSS Projekt?
2. Wie wird die Softwareentwicklung in einem OSS Projekt koordiniert?



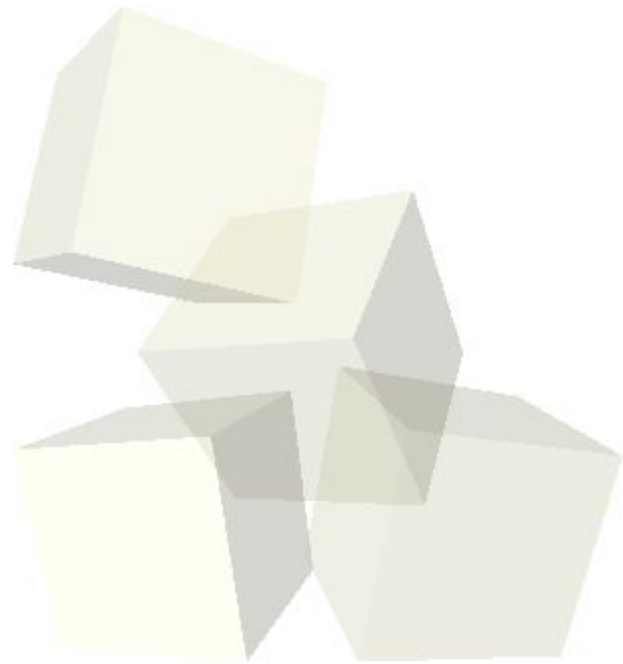


1. Wie entsteht ein OSS Projekt?
2. Wie wird die Softwareentwicklung in einem OSS Projekt koordiniert?
3. Kommunikation in OSS Projekten



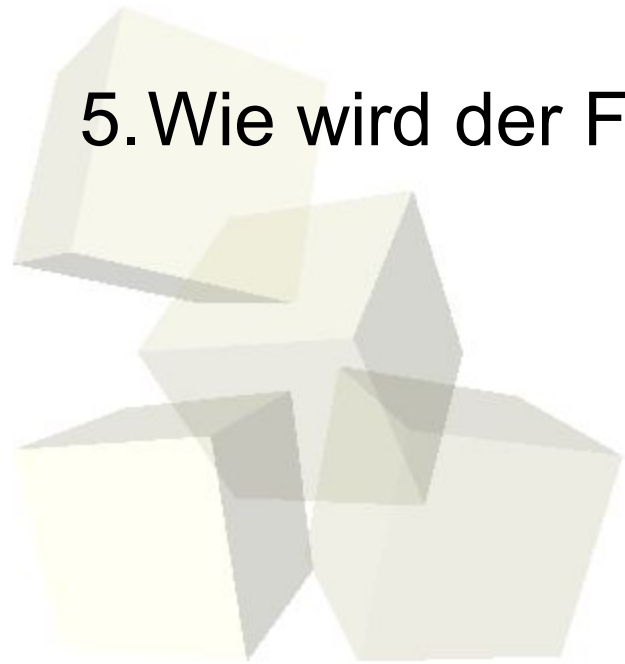


1. Wie entsteht ein OSS Projekt?
2. Wie wird die Softwareentwicklung in einem OSS Projekt koordiniert?
3. Kommunikation in OSS Projekten
4. Wie unterscheidet sich OSS von kommerzieller Softwareentwicklung?





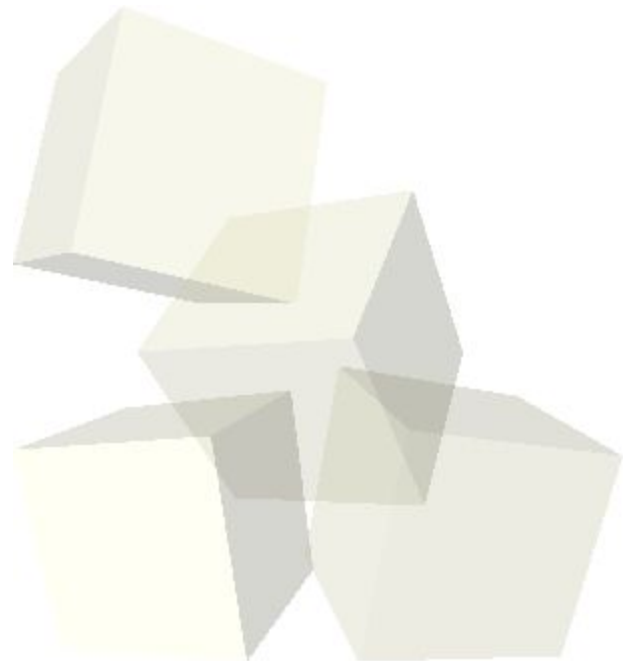
1. Wie entsteht ein OSS Projekt?
2. Wie wird die Softwareentwicklung in einem OSS Projekt koordiniert?
3. Kommunikation in OSS Projekten
4. Wie unterscheidet sich OSS von kommerzieller Softwareentwicklung?
5. Wie wird der Fortbestand eines OSS Projektes gesichert?





1. Wie entsteht ein OSS Projekt?

1. Neuentwicklung
2. "Fork" eines OSS Projektes
3. "Öffnung" einer kommerziellen Codebasis

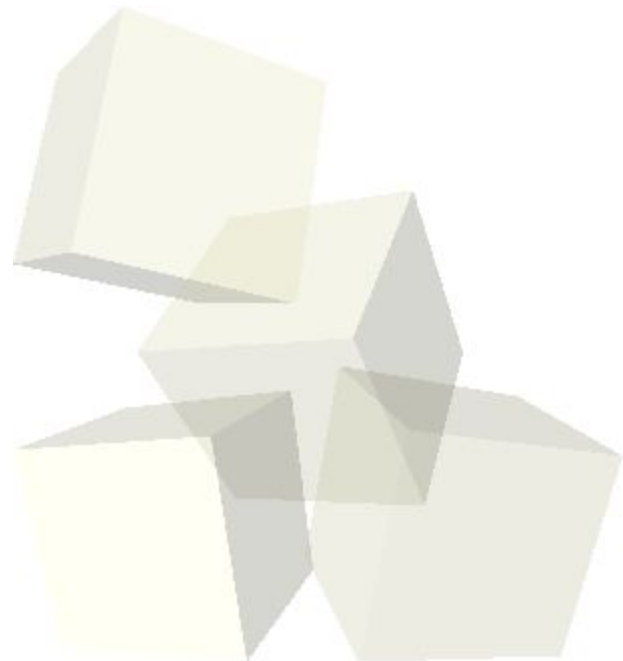




1.1. Neuentwicklung

1. Motivation für ein neues OSS Projekt:

- Es existiert noch keine Software um ein bestimmtes Problem zu lösen
- Existierende Software nur unzureichend, läuft nur auf wenigen Plattformen oder ist durch ihre Lizenz nur eingeschränkt verwendbar





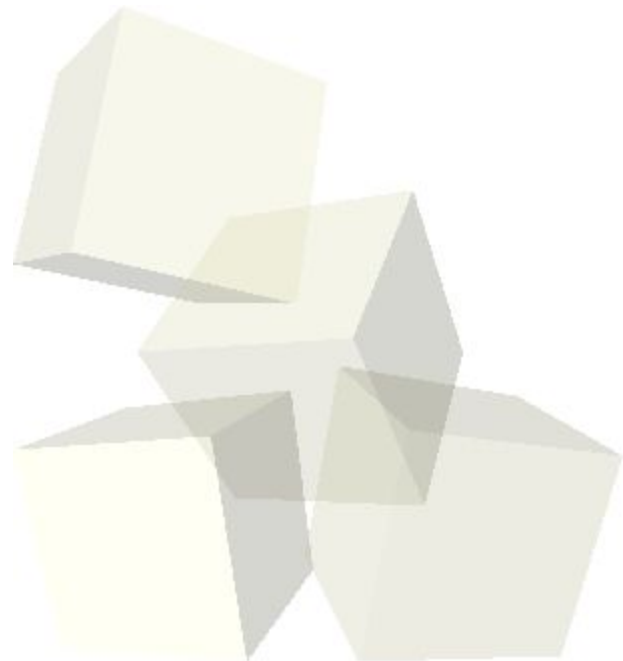
1.1. Neuentwicklung

1. Motivation für ein neues OSS Projekt:

- Es existiert noch keine Software um ein bestimmtes Problem zu lösen
- Existierende Software nur unzureichend, läuft nur auf wenigen Plattformen oder ist durch ihre Lizenz nur eingeschränkt verwendbar

2. Voraussetzungen:

- Die erste Version sollte zumindest eingeschränkt lauffähig sein
- Weiterentwicklung sollte parallel verlaufen können

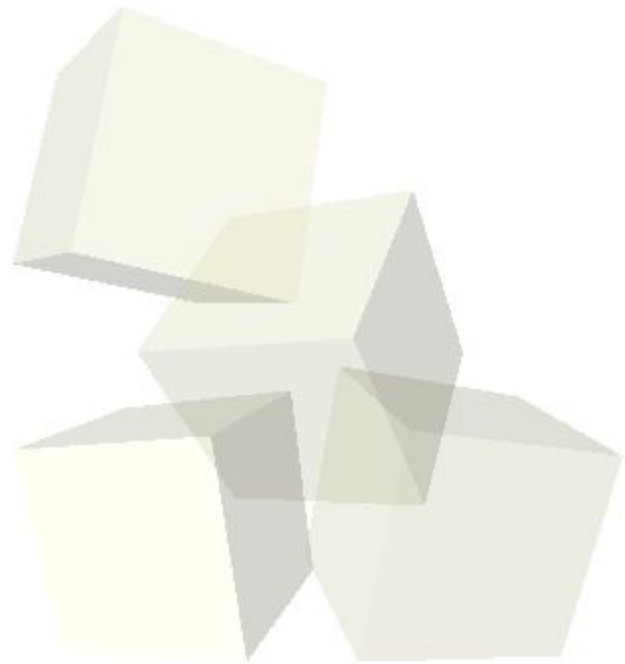




1.2. “Fork” eines OSS Projektes

1. Das Projekt wurde von den Entwicklern verlassen

- Beispiel: FreeS/WAN -> Openswan





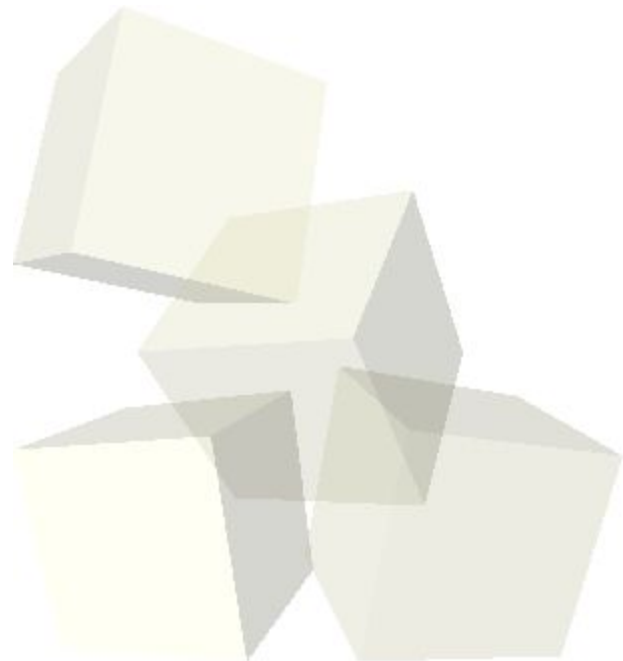
1.2. “Fork” eines OSS Projektes

1. Das Projekt wurde von den Entwicklern verlassen

- Beispiel: FreeS/WAN -> Openswan

2. Entwicklung geschieht zu langsam oder Entwickler werden von der Mitarbeit ausgeschlossen

- Beispiel: XFree86 -> KDrive





1.2. “Fork” eines OSS Projektes

1. Das Projekt wurde von den Entwicklern verlassen

- Beispiel: FreeS/WAN -> Openswan

2. Entwicklung geschieht zu langsam oder Entwickler werden von der Mitarbeit ausgeschlossen

- Beispiel: XFree86 -> KDrive

3. Ein Teil der Entwickler ist mit Entscheidungen der Projektleitung nicht einverstanden (z.B. Lizenzänderung)

- Beispiel: WINE -> ReWind



1.3. aus einer kommerziellen Codebasis

1. Mozilla

- 1998 von Netscape-Mitarbeitern gestartet
- Quellcode der unfertigen Netscape Communicator Version 5.0
- Alle Netscape-Versionen ab 6.0 aus dem Mozilla Code erstellt



1.3. aus einer kommerziellen Codebasis

1. Mozilla

- 1998 von Netscape-Mitarbeitern gestartet
- Quellcode der unfertigen Netscape Communicator Version 5.0
- Alle Netscape-Versionen ab 6.0 aus dem Mozilla Code erstellt

2. OpenOffice

- Quellcode von StarOffice Juni 2000 freigegeben
- Alle StarOffice-Versionen ab 6.0 aus dem OpenOffice Code erstellt



1.3. aus einer kommerziellen Codebasis

1. Mozilla

- 1998 von Netscape-Mitarbeitern gestartet
- Quellcode der unfertigen Netscape Communicator Version 5.0
- Alle Netscape-Versionen ab 6.0 aus dem Mozilla Code erstellt

2. OpenOffice

- Quellcode von StarOffice Juni 2000 freigegeben
- Alle StarOffice-Versionen ab 6.0 aus dem OpenOffice Code erstellt

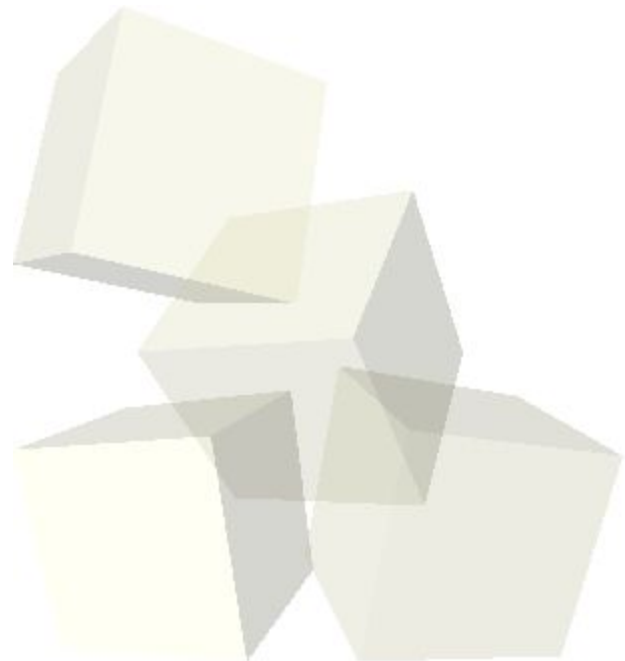
3. Blender

- Kommerzielle Entwicklung 2001 von NaN (Not a Number) eingestellt
- Spendenaktion um \$100.000 für die Freigabe zu sammeln
- 7 Wochen später Kontostand von \$100.000 überschritten



2. Softwareentwicklung in OSS Projekten

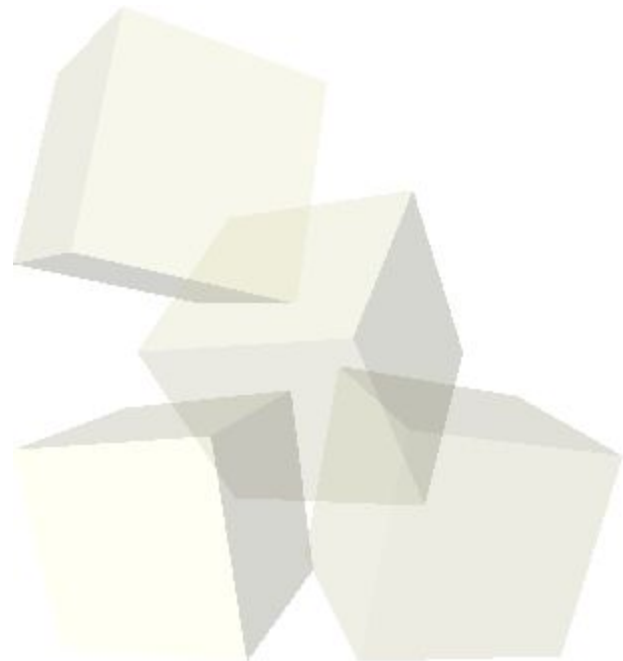
1. OSS allgemein
2. Versionskontrolle mit CVS
3. Linux
4. Mozilla
5. Debian





1. Produzierter Code hat größeren Einfluss als Geld

- Bei Beschwerden gilt im Allgemeinen: “put up or shut up”



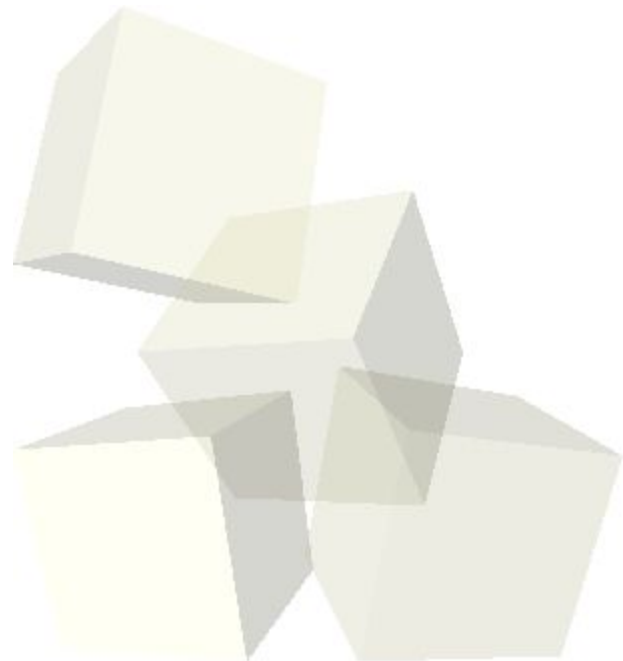


2.1. OSS allgemein

1. Produzierter Code hat größeren Einfluss als Geld

- Bei Beschwerden gilt im Allgemeinen: “put up or shut up”

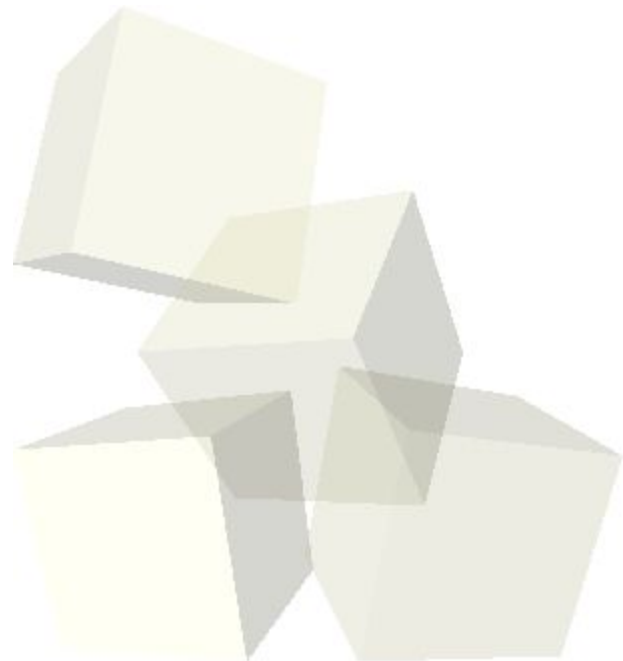
2. Softwareentwicklung meist dezentral und transparent





2.1. OSS allgemein

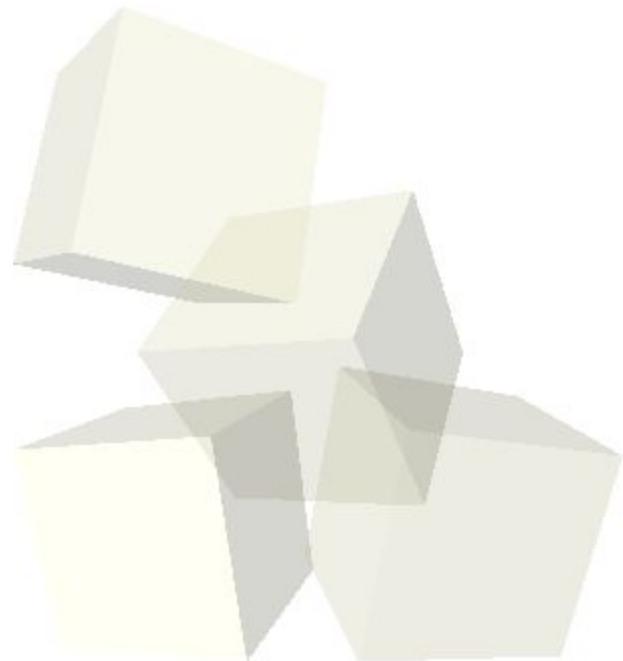
1. Produzierter Code hat größeren Einfluss als Geld
 - Bei Beschwerden gilt im Allgemeinen: “put up or shut up”
2. Softwareentwicklung meist dezentral und transparent
3. Entwickler sind fast immer auch Anwender





2.1. OSS allgemein

1. Produzierter Code hat größeren Einfluss als Geld
 - Bei Beschwerden gilt im Allgemeinen: “put up or shut up”
2. Softwareentwicklung meist dezentral und transparent
3. Entwickler sind fast immer auch Anwender
4. Kommunikation zwischen den Entwicklern erfolgt fast ausschließlich über das Internet





2.1. OSS allgemein

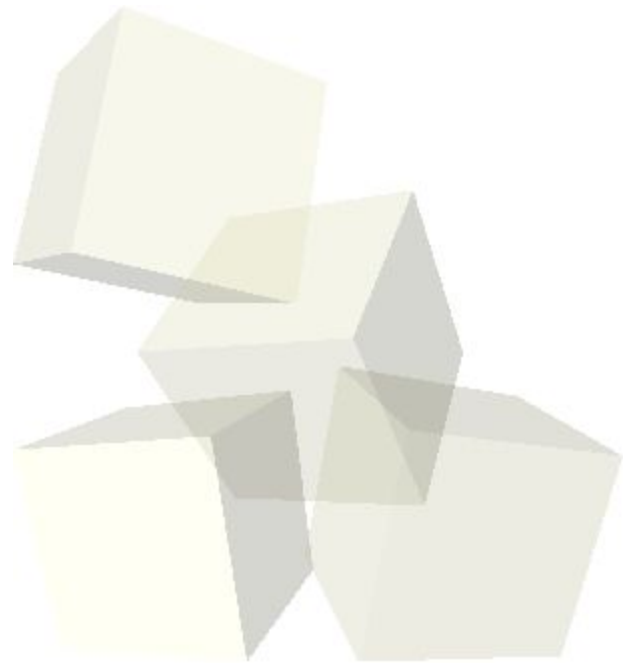
1. Produzierter Code hat größeren Einfluss als Geld
 - Bei Beschwerden gilt im Allgemeinen: “put up or shut up”
2. Softwareentwicklung meist dezentral und transparent
3. Entwickler sind fast immer auch Anwender
4. Kommunikation zwischen den Entwicklern erfolgt fast ausschließlich über das Internet
5. Vorabversionen werden veröffentlicht



2.2. Versionskontrolle mit CVS

1. Was ist CVS?

- Versionsverwaltungssystem
- Änderungen am Quelltext werden dokumentiert
- Programmierer können den Verlauf der Entwicklung überwachen

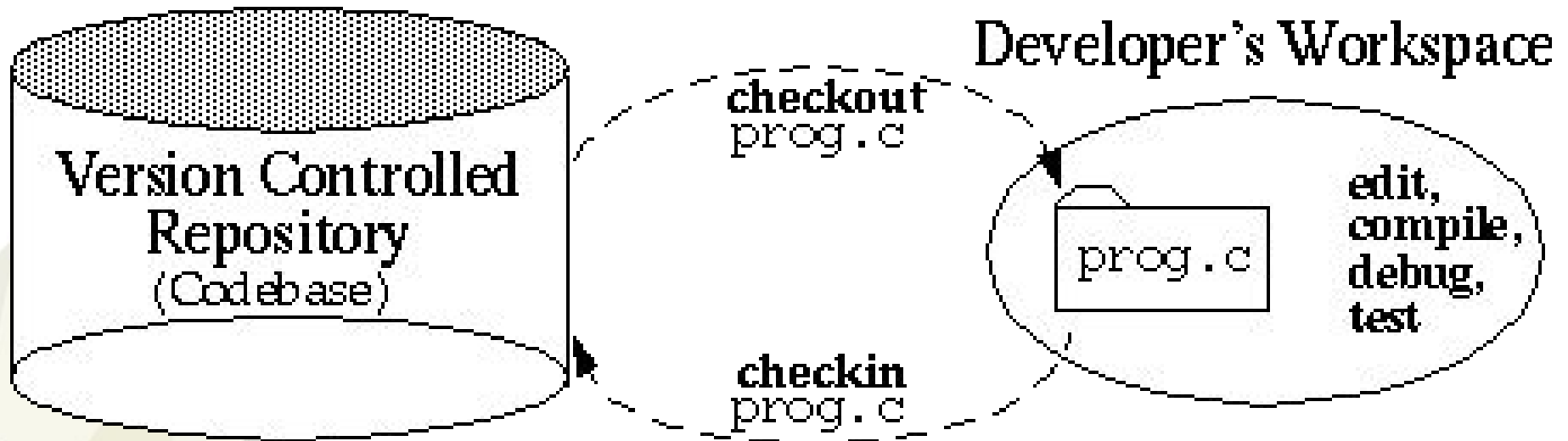




2.2. Versionskontrolle mit CVS

2. Wie funktioniert CVS?

- Checkin/Checkout

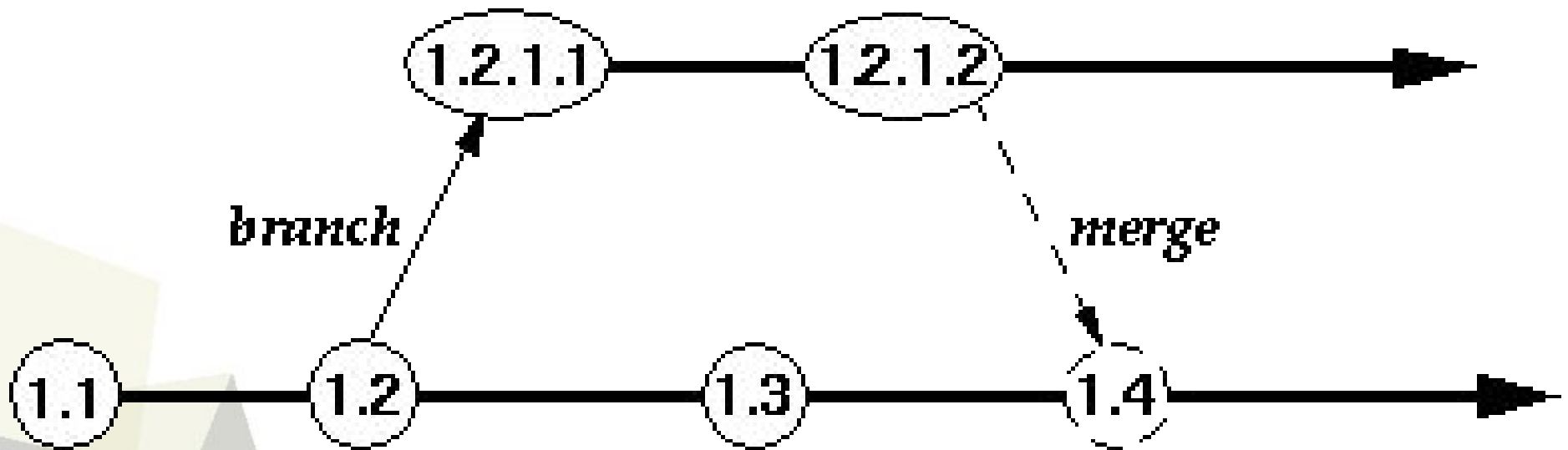




2.2. Versionskontrolle mit CVS

2. Wie funktioniert CVS?

- Checkin/Checkout
- Branching

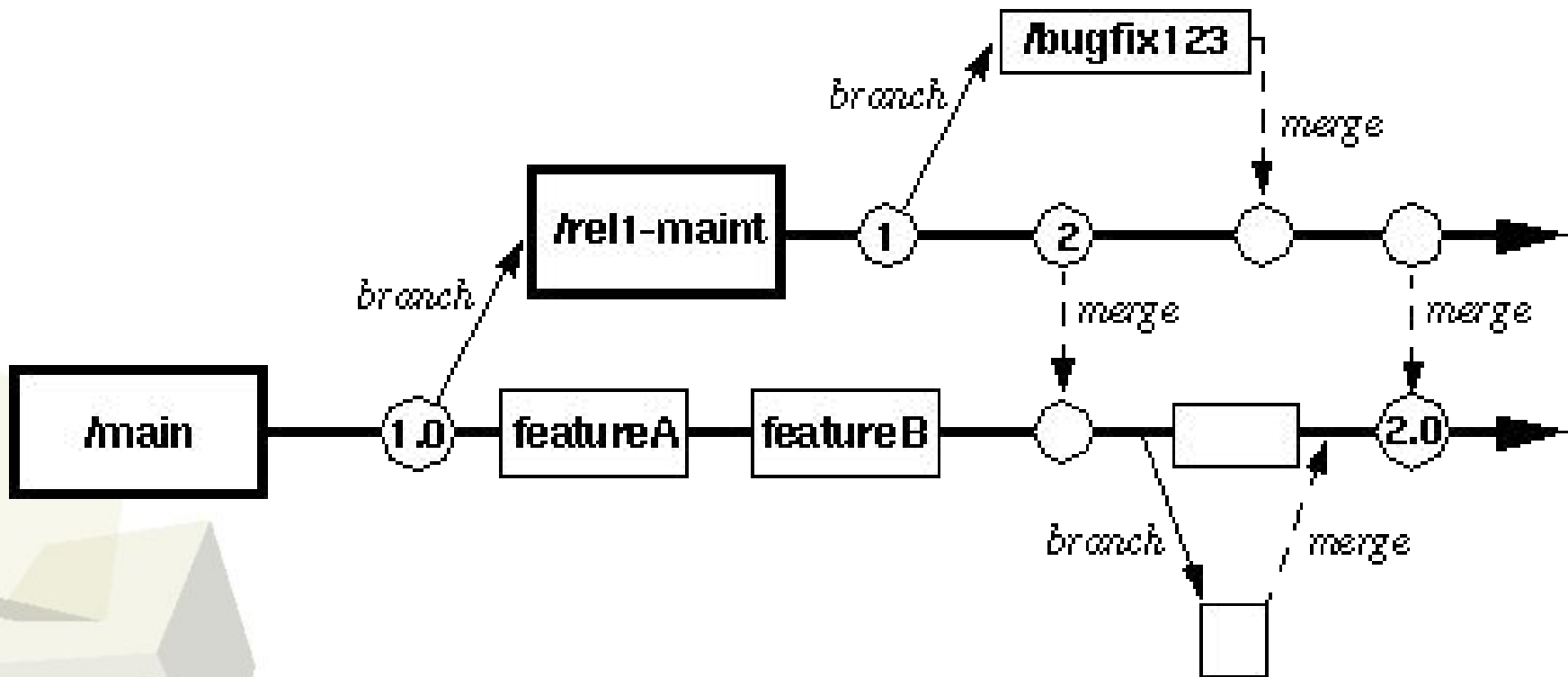




2.2. Versionskontrolle mit CVS

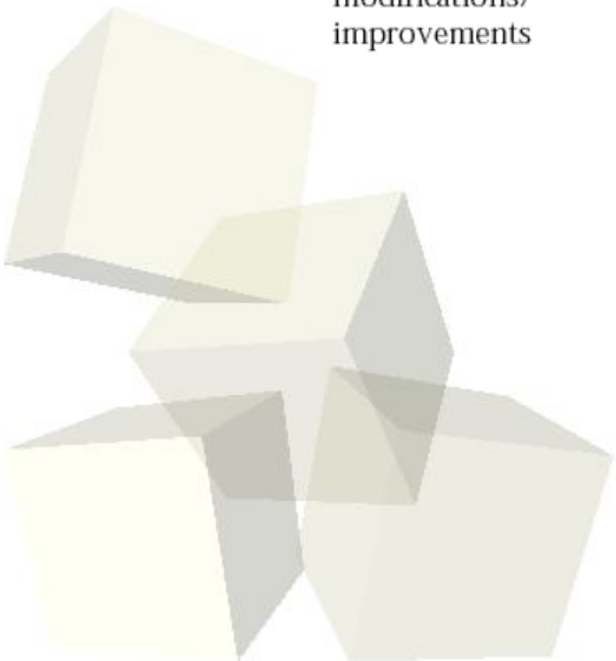
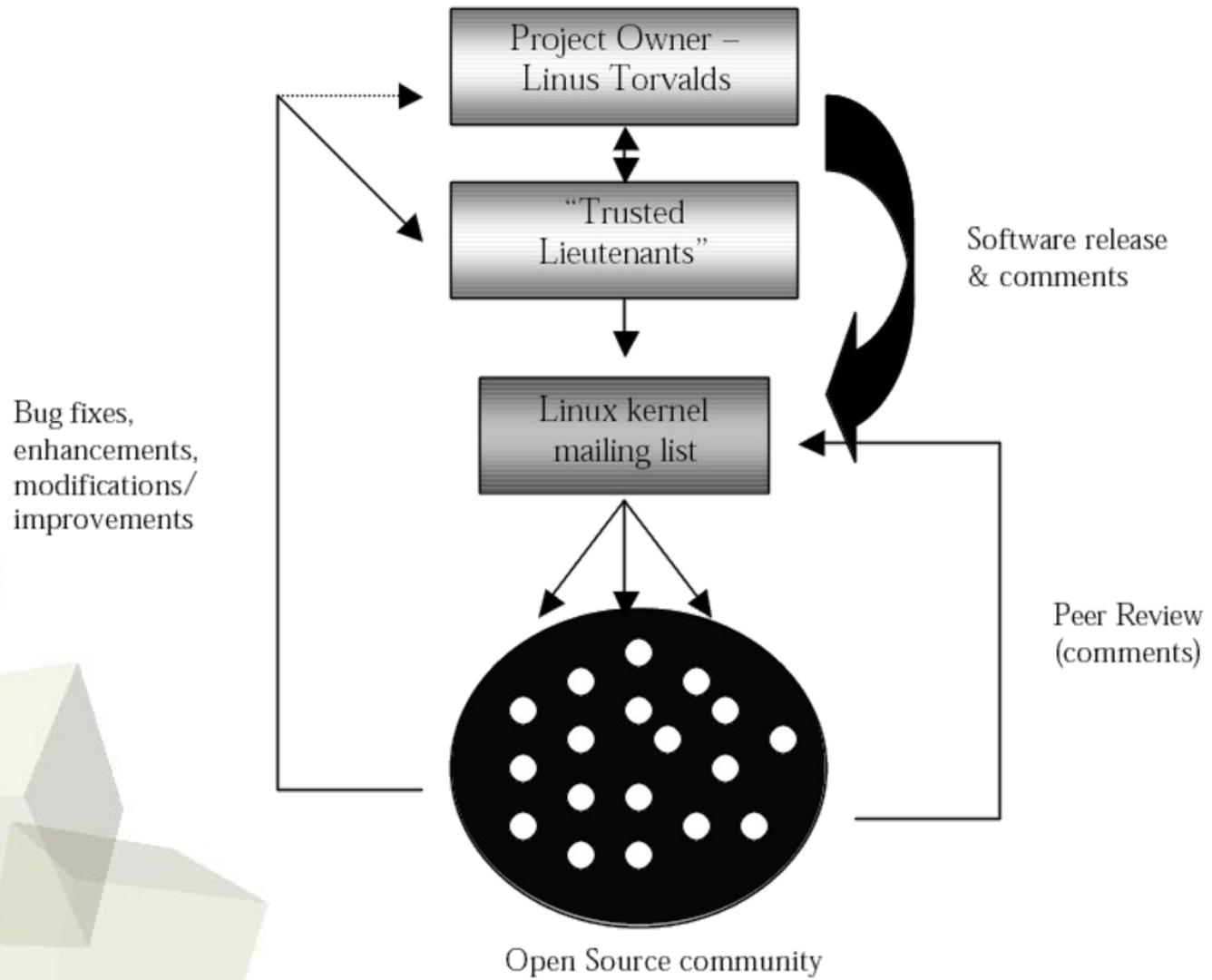
2. Wie funktioniert CVS?

- Checkin/Checkout
- Branching
- Paralleles Branching





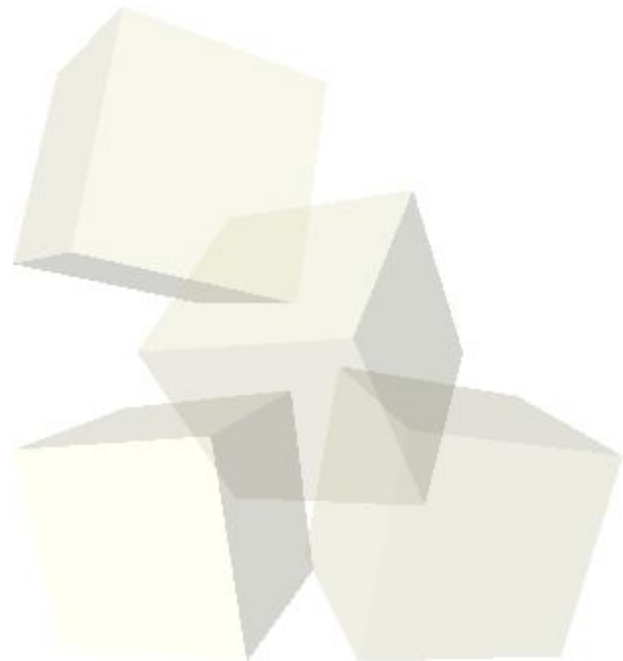
1. Projektorganisation des Linux-Kernels:





2. Nur Lesezugriff für Entwickler

- Patches von Entwicklern gehen an zuständige Maintainer
- Maintainer überprüfen den Code und leiten ihn an Linus weiter
- Nur Linus entscheidet welcher Code in den offiziellen Linux-Kernel aufgenommen wird.

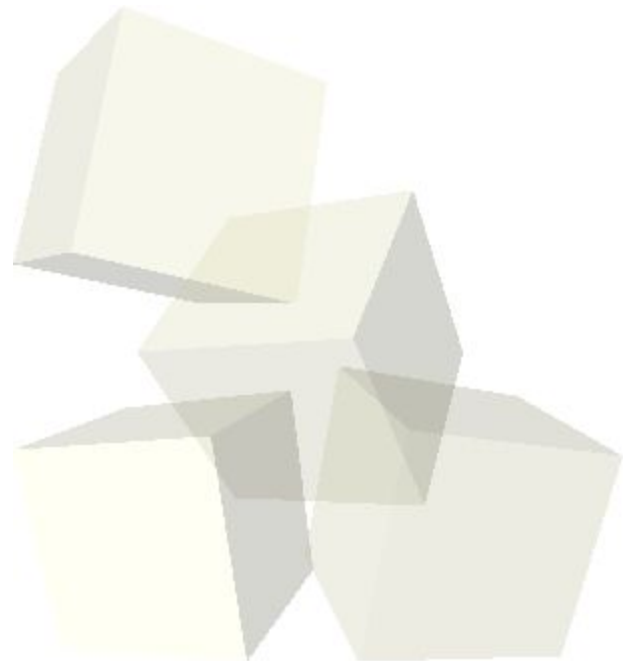




2. Nur Lesezugriff für Entwickler

- Patches von Entwicklern gehen an zuständige Maintainer
- Maintainer überprüfen den Code und leiten ihn an Linus weiter
- Nur Linus entscheidet welcher Code in den offiziellen Linux-Kernel aufgenommen wird.

3. Als Entwicklungsplattform wird BitKeeper verwendet





2. Nur Lesezugriff für Entwickler

- Patches von Entwicklern gehen an zuständige Maintainer
- Maintainer überprüfen den Code und leiten ihn an Linus weiter
- Nur Linus entscheidet welcher Code in den offiziellen Linux-Kernel aufgenommen wird.

3. Als Entwicklungsplattform wird BitKeeper verwendet

4. Zwei Versionstypen:

- Stabile Version (gerade Versionsnummern: 2.4, 2.6) für den Einsatz in Produktionsumgebungen.
- Entwicklerversion (ungerade Versionsnummern: 2.3, 2.5)



2. Nur Lesezugriff für Entwickler

- Patches von Entwicklern gehen an zuständige Maintainer
- Maintainer überprüfen den Code und leiten ihn an Linus weiter
- Nur Linus entscheidet welcher Code in den offiziellen Linux-Kernel aufgenommen wird.

3. Als Entwicklungsplattform wird BitKeeper verwendet

4. Zwei Versionstypen:

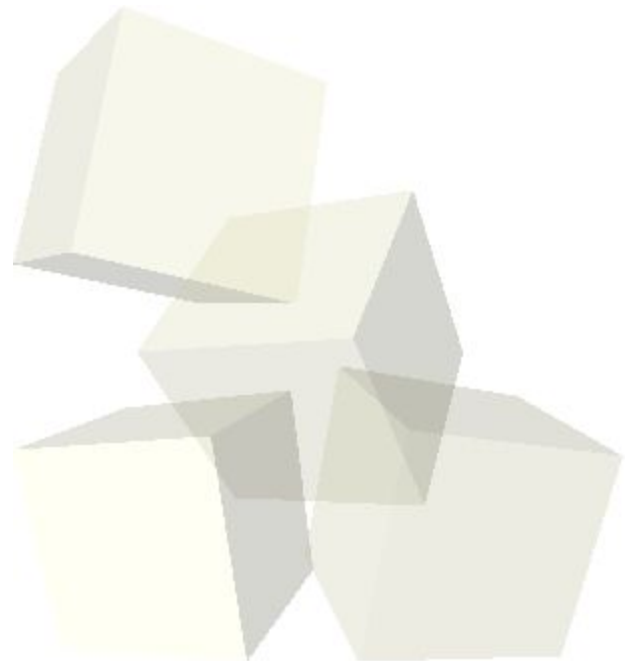
- Stabile Version (gerade Versionsnummern: 2.4, 2.6) für den Einsatz in Produktionsumgebungen.
- Entwicklerversion (ungerade Versionsnummern: 2.3, 2.5)

5. Releases werden ohne Zeitvorgaben je nach Stabilität veröffentlicht.



1. “Mozilla Foundation”

- Projektübergreifende Organisation
- Stellt die gesamte Infrastruktur bereit
- Besteht überwiegend aus Netscape-Entwicklern



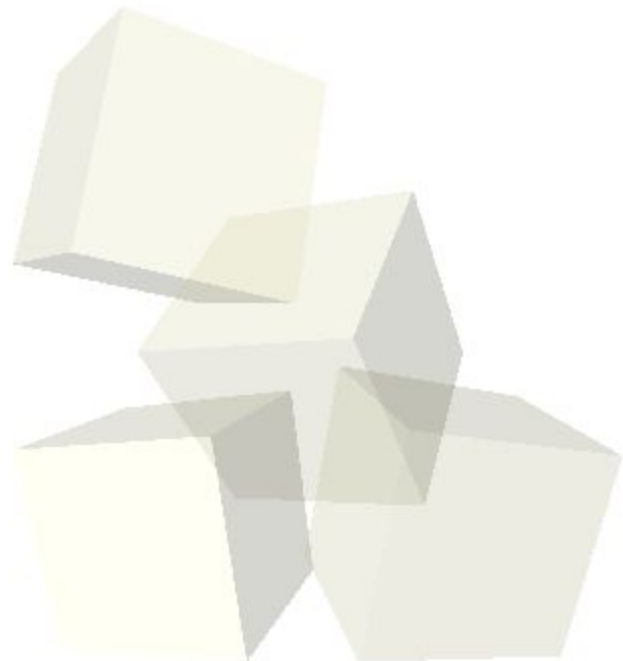


1. “Mozilla Foundation”

- Projektübergreifende Organisation
- Stellt die gesamte Infrastruktur bereit
- Besteht überwiegend aus Netscape-Entwicklern

2. “Mozilla Foundation Staff”

- Leiten Mozilla-Projekte
- Vergeben Zuständigkeitsbereiche





1. “Mozilla Foundation”

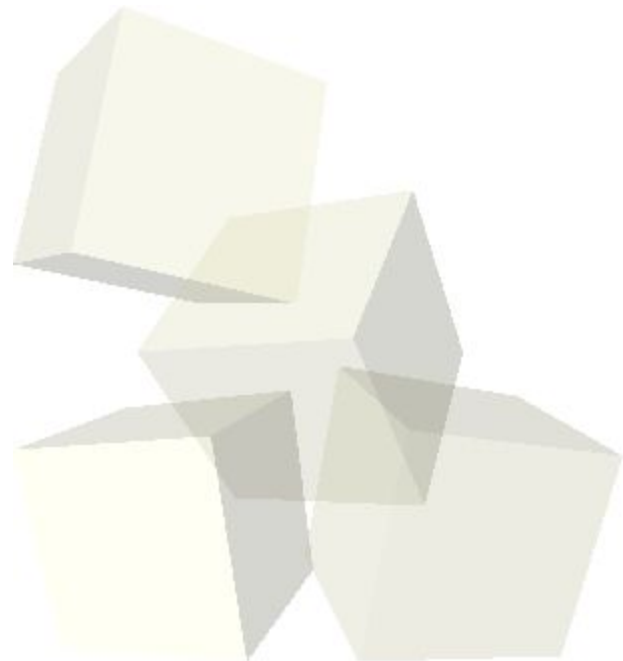
- Projektübergreifende Organisation
- Stellt die gesamte Infrastruktur bereit
- Besteht überwiegend aus Netscape-Entwicklern

2. “Mozilla Foundation Staff”

- Leiten Mozilla-Projekte
- Vergeben Zuständigkeitsbereiche

3. “Module Owners”

- Leiten die Entwicklung der einzelnen Module





1. “Mozilla Foundation”

- Projektübergreifende Organisation
- Stellt die gesamte Infrastruktur bereit
- Besteht überwiegend aus Netscape-Entwicklern

2. “Mozilla Foundation Staff”

- Leiten Mozilla-Projekte
- Vergeben Zuständigkeitsbereiche

3. “Module Owners”

- Leiten die Entwicklung der einzelnen Module

4. “Reviewers” und “Super Reviewers”

- Überprüfen Patches von der Community



1. “Mozilla Foundation”

- Projektübergreifende Organisation
- Stellt die gesamte Infrastruktur bereit
- Besteht überwiegend aus Netscape-Entwicklern

2. “Mozilla Foundation Staff”

- Leiten Mozilla-Projekte
- Vergeben Zuständigkeitsbereiche

3. “Module Owners”

- Leiten die Entwicklung der einzelnen Module

4. “Reviewers” und “Super Reviewers”

- Überprüfen Patches von der Community

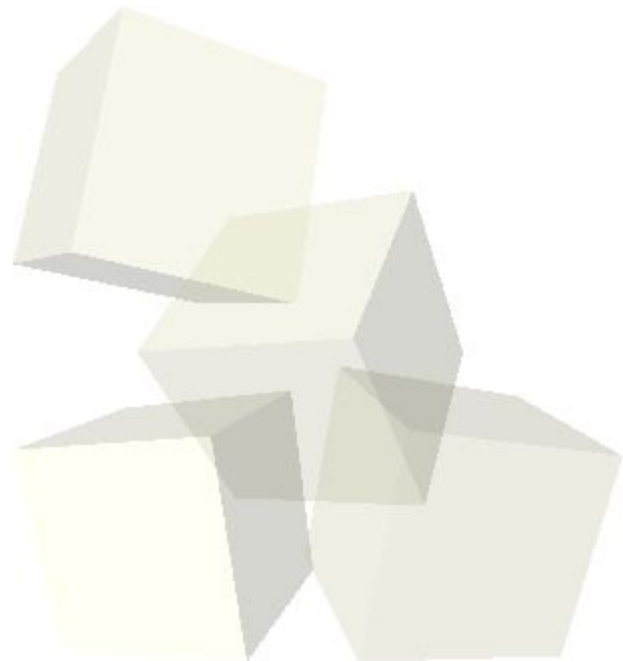
5. Infrastruktur:

- BugZilla
- CVS
- Tinderbox



6. Mehrere Projekte mit gemeinsamem Code im CVS

- Mozilla (Application Suite)
- Firefox
- Thunderbird
- Camino

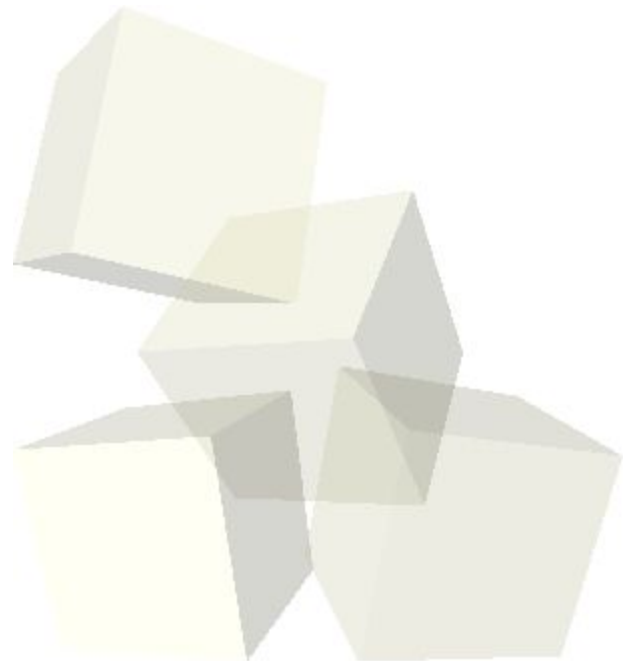




6. Mehrere Projekte mit gemeinsamem Code im CVS

- Mozilla (Application Suite)
- Firefox
- Thunderbird
- Camino

7. Bug-gesteuerte Releases





6. Mehrere Projekte mit gemeinsamem Code im CVS

- Mozilla (Application Suite)
- Firefox
- Thunderbird
- Camino

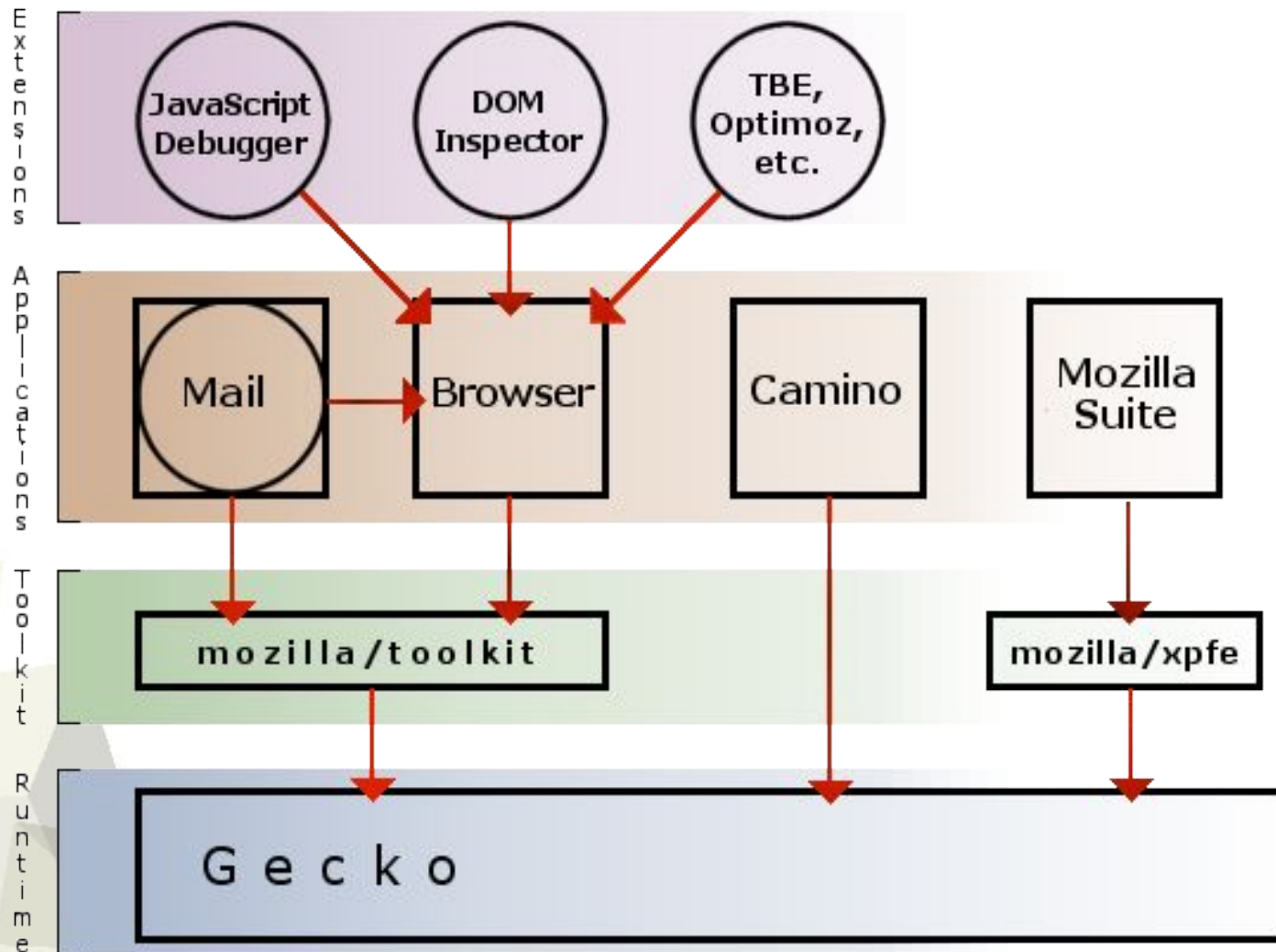
7. Bug-gesteuerte Releases

8. Zwei Versionstypen der Application Suite

- Die Referenzversion dient als Plattform für andere Anwendungen
- Die stabile Version ist für Endanwender gedacht

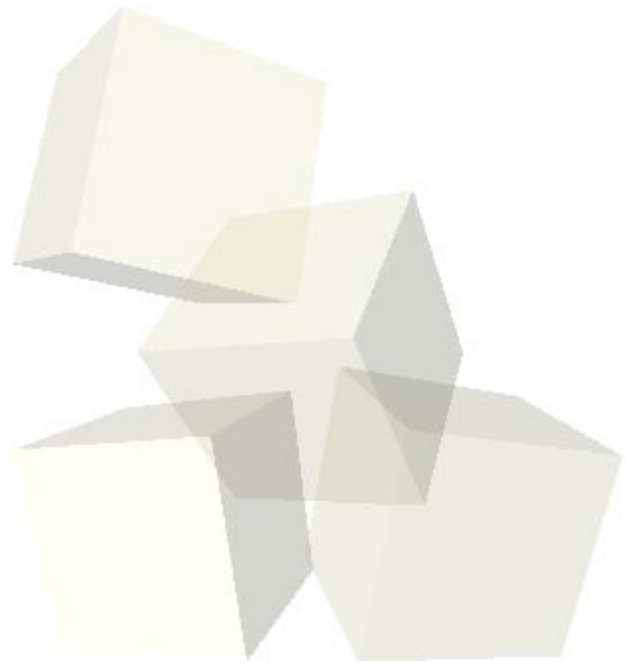


9. Architektur der Mozilla-Software





1. Komplexes System bestehend aus Verfassung und
“Social Contract”

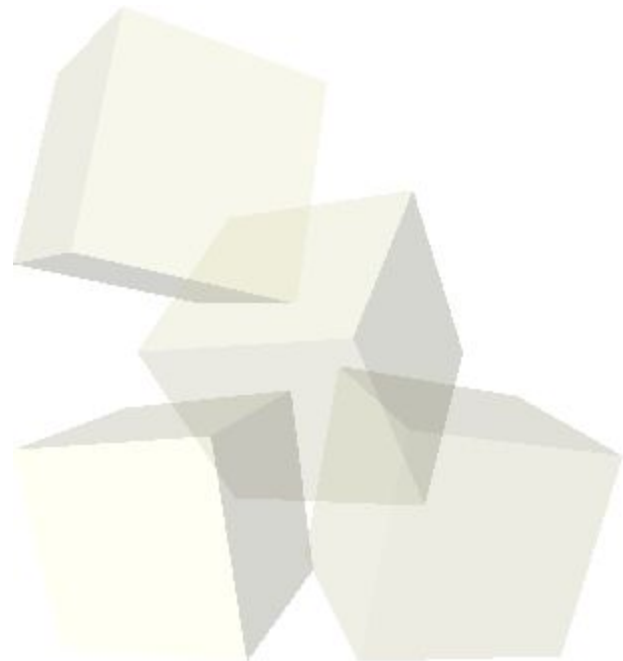




1. Komplexes System bestehend aus Verfassung und “Social Contract”

2. Flache Hierarchie

- Fast alle Entwickler haben die gleichen Rechte
- Strikte Kontrolle für Anwarter
- Demokratische Wahlen





1. Komplexes System bestehend aus Verfassung und “Social Contract”

2. Flache Hierarchie

- Fast alle Entwickler haben die gleichen Rechte
- Strikte Kontrolle für Anwarter
- Demokratische Wahlen

3. Ämter in der Organisation

- Projektleiter
 - offizieller Repräsentant
- Schriftführer
 - Wahlen
 - vertritt andere Amtsinhaber
 - Auslegung der Verfassung
- Technischer Ausschuss
 - entscheidet endgültig bei technischen Diskussionen



1. Komplexes System bestehend aus Verfassung und “Social Contract”

2. Flache Hierarchie

- Fast alle Entwickler haben die gleichen Rechte
- Strikte Kontrolle für Anwärter
- Demokratische Wahlen

3. Ämter in der Organisation

- Projektleiter
 - offizieller Repräsentant
- Schriftführer
 - Wahlen
 - vertritt andere Amtsinhaber
 - Auslegung der Verfassung
- Technischer Ausschuss
 - entscheidet endgültig bei technischen Diskussionen

4. Hauptarchiv nur aus Freier Software

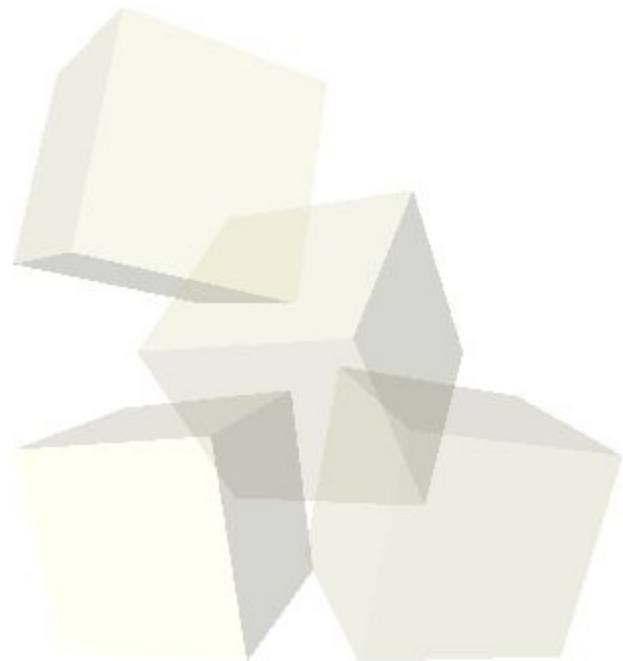
- “contrib” und “non-free” für andere Software
- Aufnahmekriterien für Pakete (Debian Policy)



3. Kommunikation in OSS Projekten

1. CVS

- Logs dokumentieren jeden Checkin
- Einzelne Checkins Ausgangspunkte für Diskussionen



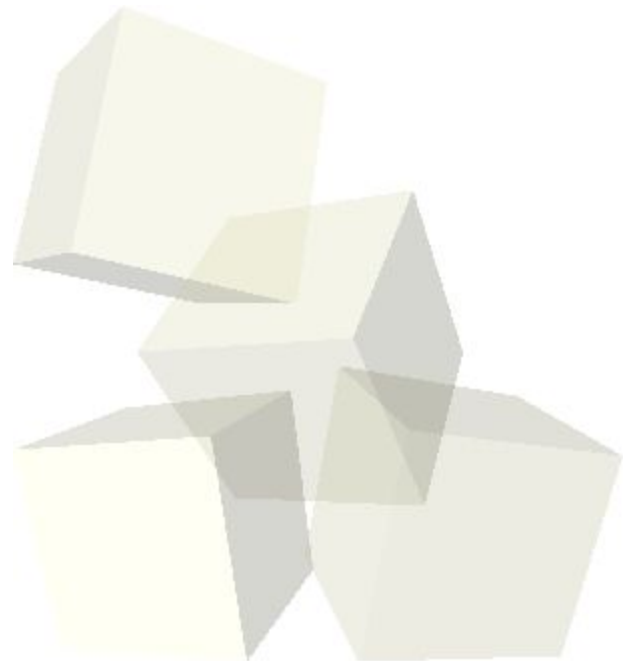


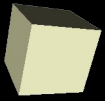
3. Kommunikation in OSS Projekten

1. CVS

- Logs dokumentieren jeden Checkin
- Einzelne Checkins Ausgangspunkte für Diskussionen

2. Mailing List





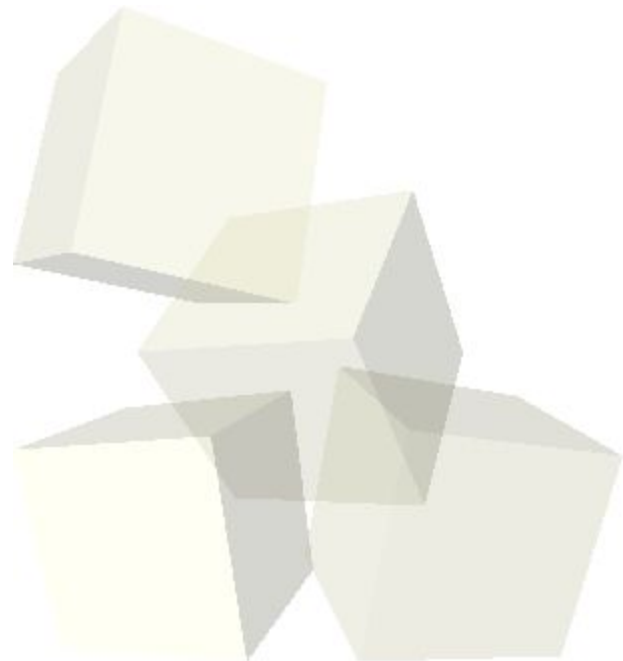
3. Kommunikation in OSS Projekten

1. CVS

- Logs dokumentieren jeden Checkin
- Einzelne Checkins Ausgangspunkte für Diskussionen

2. Mailing List

3. IRC





3. Kommunikation in OSS Projekten

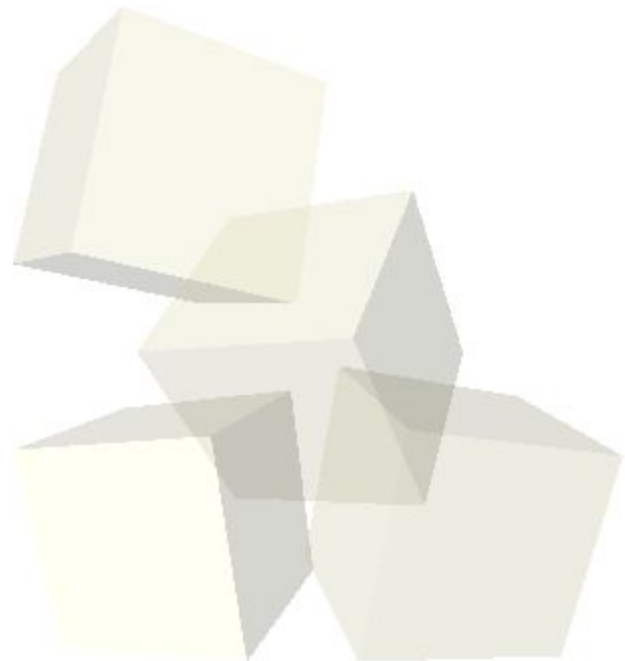
1. CVS

- Logs dokumentieren jeden Checkin
- Einzelne Checkins Ausgangspunkte für Diskussionen

2. Mailing List

3. IRC

4. Bug Tracking System

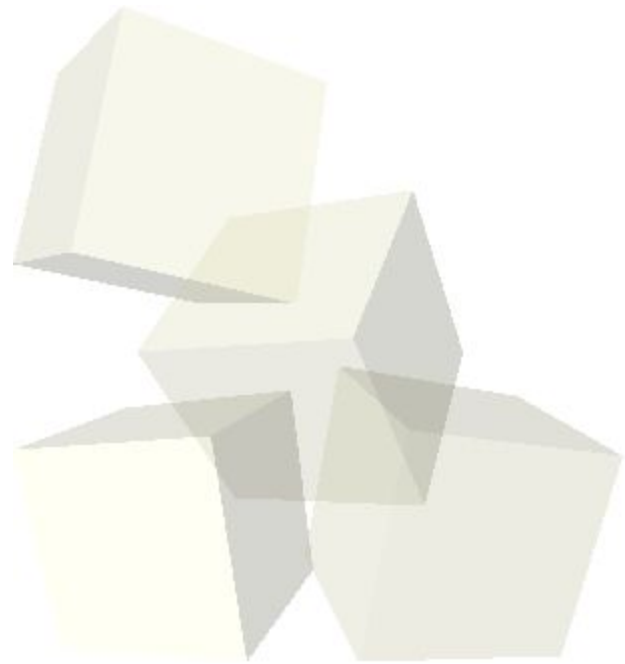




4. Kommerzielle Softwareentwicklung

1. Organisation

- Meist striktere Rangordnungen
- Management entscheidet über Softwareentwicklung
- Release-Daten müssen eingehalten werden





4. Kommerzielle Softwareentwicklung

1. Organisation

- Meist striktere Rangordnungen
- Management entscheidet über Softwareentwicklung
- Release-Daten müssen eingehalten werden

2. Marketing

- Höherer Stellenwert, da das Produkt verkauft werden muss
- Bei OSS zu vernachlässigen
 - Gute Software spricht sich herum
 - Index-Seiten (wie z.B. <http://freshmeat.net>)





4. Kommerzielle Softwareentwicklung

1. Organisation

- Meist striktere Rangordnungen
- Management entscheidet über Softwareentwicklung
- Release-Daten müssen eingehalten werden

2. Marketing

- Höherer Stellenwert, da das Produkt verkauft werden muss
- Bei OSS zu vernachlässigen
 - Gute Software spricht sich herum
 - Index-Seiten (wie z.B. <http://freshmeat.net>)

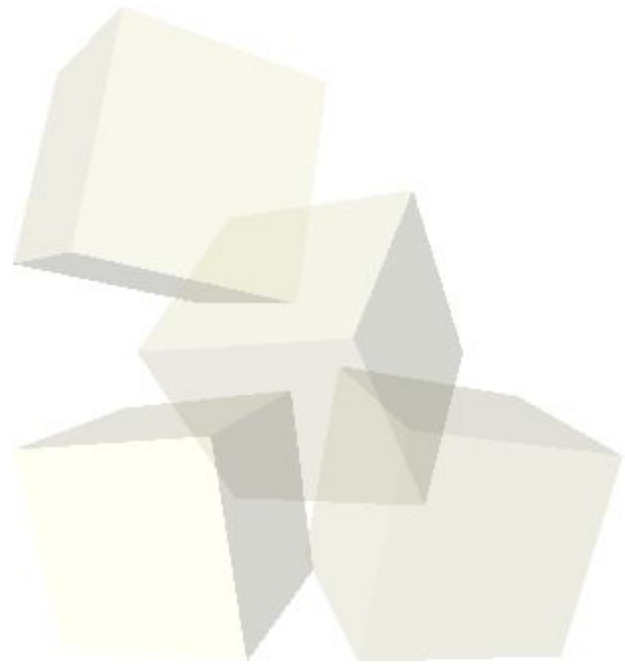
3. Entwicklungskosten

- Spielen eine große Rolle, da die Programmierer bezahlt werden
- Können zum vorzeitigen Abbruch führen
- Bei OSS zu vernachlässigen



5. Fortbestand eines OSS Projektes

1. Motivation der Entwickler



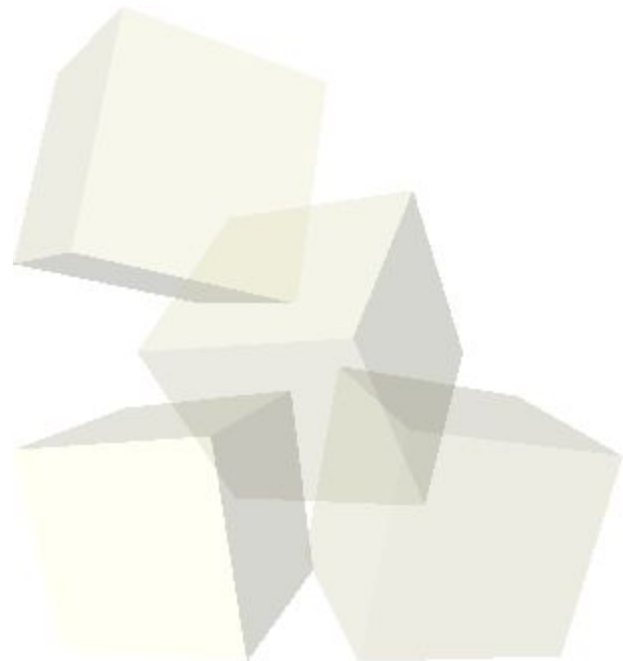


5. Fortbestand eines OSS Projektes

1. Motivation der Entwickler

2. Popularität der Software

- Projekt muss die “kritische Masse” erreicht haben
- Obsolete Software wird normalerweise nicht mehr weiterentwickelt





5. Fortbestand eines OSS Projektes

1. Motivation der Entwickler

2. Popularität der Software

- Projekt muss die “kritische Masse” erreicht haben
- Obsolete Software wird normalerweise nicht mehr weiterentwickelt

3. Unterstützung durch Firmen

- Nicht essenziell, kann aber zum Erfolg beitragen

