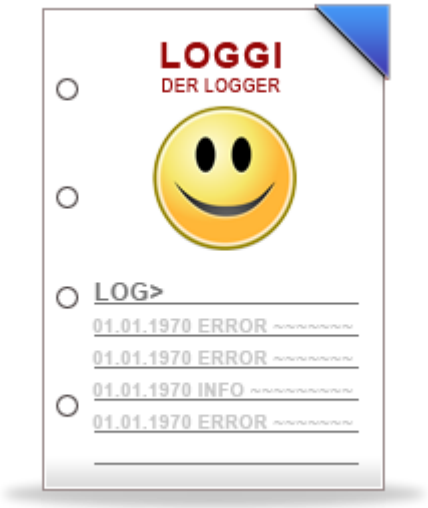

Kundenanforderungen

Hochschule Luzern
Technik & Architektur

Software Komponenten – FS13

Gruppe 03 | Horw, 24.05.2013
Bontekoe Christian | Estermann Michael | Moor Simon | Rohrer Felix

Autoren

Bontekoe Christian	
Studiengang	Informatiker (Berufsbegleitend)
Adresse	
Telefon	
E-Mail	

Estermann Michael	
Studiengang	Informatiker (Berufsbegleitend)
Adresse	
Telefon	
E-Mail	

Moor Simon	
Studiengang	Informatiker (Berufsbegleitend)
Adresse	
Telefon	
E-Mail	

Rohrer Felix	
Studiengang	Informatiker (Berufsbegleitend)
Adresse	
Telefon	
E-Mail	

Änderungskontrolle

Version	Datum	Autor	Beschreibung
1.0	15.03.2013	Felix Rohrer	Vorlage erstellen, div. Texte erstellen
1.0	16.04.2013	Felix Rohrer	Version 1.0 freigegeben
1.1	19.04.2013	Christian Bontekoe	Anpassungen gemäss Review
1.2	25.04.2013	Simon Moor	Corba hinzugefügt
2.0	24.05.2013	Christian Bontekoe	Version 2.0 freigegeben

Inhalt

1 Ziel

Verschiedene Betriebssysteme stellen integrierte Logging-Mechanismen zur Verfügung: Unixbasierende Systeme z.B. das so genannte 'Syslog' und Windows-basierende Systeme das 'EventLog'. Im Rahmen dieser Aufgabe soll ein alternatives und komponentenbasierendes Message-Loggingsystem entwickelt werden, welches unabhängig von der Plattform und auch auf mehrere Hosts verteilt verwendet werden kann.

2 Referenzen

- SWK Aufgabenstellung (SWK_Aufgabenstellung-V10 Aktionen.pdf)
- SWK Aufgabenstellung Teil 2 (SWK_Aufgabenstellung_CORBA_10 Aktionen)

3 Produktübersicht

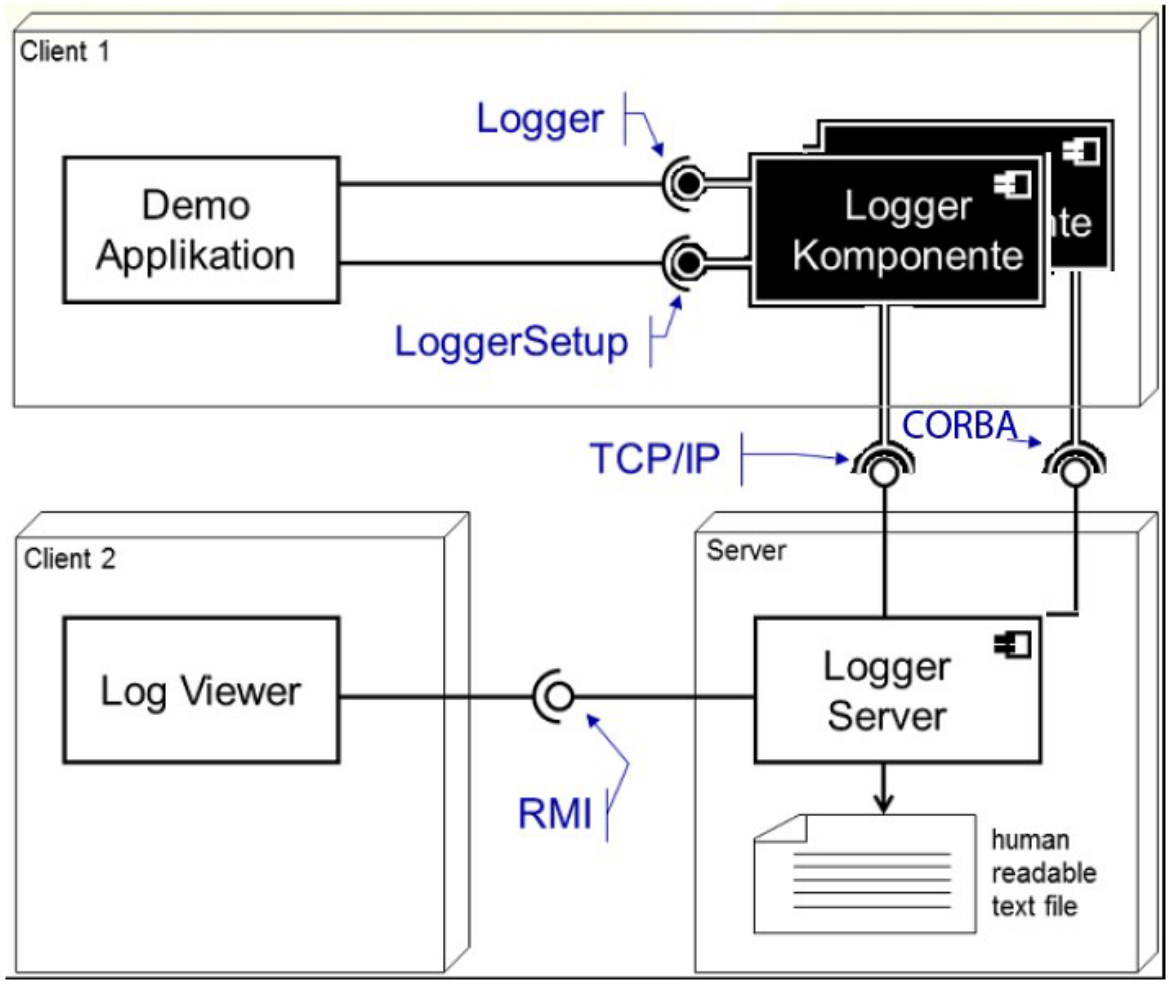

Abbildung 1 Produktübersicht

- Logger-Server
  Speichern der Einträge, weiterleiten von Meldungen an Logger-Viewer
  - TCP-Server
    Clients verbinden via Logger-Komponente
  - RMI-Server
    Kommunikation mit dem Logger-Viewer
  - Corba-Server
    Logger-Komponente verbindet via Corba
- Logger-Client
  Verbinden auf Server via Logger-Komponente, Senden von Logs an Server
- Logger-Komponente
  Verbinden auf Logger-Server, weiterleiten von Messages an Server
  Nebst TCP-IP wird ebenfalls Corba als Verbindungsmöglichkeit akzeptiert
- Log-Viewer
  Darstellen der Log Einträge auf dem Bildschirm

4 Produkt-Funktionen

4.1 Grundfunktionalität des Message-Loggers

Bei der Logger-Komponente handelt es sich um eine Software-Komponente, die sehr einfach in bestehende Java-Applikationen eingebunden werden kann. Eine Applikation kann durch einfache Methodenaufrufe textbasierende Meldungen (Messages) mit ihrer Logger-Komponente dauerhaft aufzeichnen. Die Meldungen aller Applikationen sollen in einem zentralen Logger Server in einem vordefinierten Format gespeichert werden.

Mit einem Log-Viewer, einer eigenständigen Anwendung, die unabhängig von der LoggerKomponente arbeitet, können die auf dem Server eintreffenden Messages (einer oder mehrerer Applikationen) online verfolgt und angezeigt werden. Online bedeutet, dass Meldungen sofort und ohne Benutzerinteraktion angezeigt werden, sobald eine neue Message auf dem Server eintrifft.

4.2 Einsatzgebiete

Das Message-Loggingsystem kann z.B. für Debug-Zwecke (Tracing, die Applikation meldet durch Meldungseinträge den zurückgelegten Programmfluss und Zwischenresultate) oder für das ErrorLogging im Feld (die Applikation meldet wichtige Fehlersituationen und notwendige Information dazu) verwendet werden.

5 Produktdaten

Muss-Features

Nr.	Bezeichnung
1	Die Logger-Komponente muss als Komponente (mit Interface) realisiert werden.
2	Die dauerhafte Speicherung der Messages erfolgt in einem einfachen Textfile (welches somit auch mit ‚Bordmitteln' des jeweiligen Betriebssystems eingesehen werden kann). Das File enthält mindestens die Quelle der Logmeldung, einen Zeitstempel, den Message-Level und den Message-Text.
3	Die Logger-Komponente ist austauschbar und plattformunabhängig zu realisieren. Der Komponentenaustausch muss ausserhalb der Entwicklungsumgebung und ohne CodeAnpassung, d.h. ohne Neukompilation möglich sein.
4	Es muss möglich sein, dass mehrere Applikationen über ihre jeweilige Instanz der LoggerKomponente parallel auf den zentralen Message-Server loggen.
5	Log-Viewer: Unabhängige Anwendung, welche sowohl bereits früher aufgezeichnete Einträge als auch die aktuellen Einträge online anzeigt. Die Kommunikation soll dabei über RMI erfolgen (RMI = Remote Method Invocation).
6	Bei jedem Log-Eintrag hat die aufrufende Applikation einen Message-Level mitzugeben (z.B.: Fehler, wichtige Meldung, normale Meldung, unwichtige Meldung). Über die API der LoggerKomponente kann ein Level-Filter gesetzt werden. Damit kann definiert werden, welche Meldungen (mit welchem Level) tatsächlich übertragen werden. Dieser Level kann zur Laufzeit geändert werden.
8	Die Logger-Komponente benötigt zwei Software-Interfaces: Logger: Message erzeugen und eintragen. Eine Applikation kann via Methodenaufruf Messages (Textstrings) loggen. LoggerSetup: Dient zur Konfiguration des Message Loggers, z.B. setLevel(…) Weitere Interfaces sind wo sinnvoll individuell zu definieren.
9	Statische Konfigurationsdaten der Komponenten (z.B.: Informationen bezüglich Erreichbarkeit des Servers) sind zu definieren und müssen ohne Programmierung anpassbar sein.

6 Qualitätsanforderungen, Leistungsanforderungen

Dieses Projekt muss vollständig in Java entwickelt werden. Es muss in den vorgegebenen Modulen in Subversion verwaltet und auf dem Buildserver laufend integriert werden.

Es geht darum, praktische Erfahrung bei Projektmanagement, Spezifikation, Design, Implementierung, Test, Dokumentation und Deployment eines komponentenbasierten Systems zu sammeln.

- Die Planung muss für ein iteratives Vorgehen mit mindestens zwei Iterationen aufgesetzt werden. Es ist davon auszugehen, dass ca. in der 8. Semesterwoche eine Zwischenabgabe (Prototyp) erfolgt und in der 9. Semesterwoche vom Dozierendenteam zusätzliche Vorgaben gemacht werden, die eine Überarbeitung und einen neuen Release des Message-LoggerSystems erfordern.
- Spezifikation und Design sind mittels der im Selbststudium (UML Kurzreferenz, Oestereich) erarbeiteten UML-Elemente zu beschreiben.
- Die Implementierung soll nur auf den Java Standard-Bibliotheken aufbauen.
- Mindestens für die Klassen der Logger-Komponente und des Logger-Servers sind JUnitTests Teil der geforderten Abgabe. Unit-Tests sind aber generell empfohlen.

Der Code soll so aufgebaut sein, dass er folgende Punkte zulässt:

- Erweiterbarkeit
- Wartbarkeit
- Übertragbarkeit
- Wiederverwendbarkeit

Abbildungsverzeichnis