

Alpha & ooo Ltd.



Log Analyzer

Version: 1.0

Date: 24.11.2008

Inhalt

Einleitung	4
Funktionen	4
Systemübersicht	4
<i>Subsysteme</i>	5
<i>Subsystem – Reader</i>	5
Programmkurzbeschreibung	5
Datenorganisation/Voraussetzungen	5
Funktionshierarchie	6
Fehlerbehandlung	6
Anwendung	6
Programmaufbau	6
Programmablauf	7
<i>Subsystem - Analyzer</i>	9
Programmkurzbeschreibung	9
Datenorganisation/Voraussetzungen	10
Funktionshierarchie	10
Fehlerbehandlung	11
Anwendung	11
Programmaufbau	11
Programmablauf	12
Konfiguration	13
<i>Subsystem - WebFrontend</i>	13
Programmkurzbeschreibung	13
Funktionshierarchie	14
Fehlerbehandlung	14
Programmaufbau	14
Programmablauf	15
Konfiguration	16
<i>Subsystem - Datenbank</i>	16
Datenbankname	16
Datenbanksystem	16
Programminstallation	17
<i>Softwarebedarf</i>	17
<i>Installationsanweisung</i>	17
Installation Web-Monitor	18
Installation Reader	18
Installation Analyzer	18
Programmaufbau	19

Abbildungsverzeichnis

Abbildung 1 Gesamtsystem	4
Abbildung 2 Subsysteme	5
Abbildung 3 Funktionshierarchie LAReader	6
Abbildung 4 Programmbausteine LAReader	6
Abbildung 5 Programmablauf LAReader	7
Abbildung 6 Programmfunktion LAAalyzer	9
Abbildung 7 Programmfunktion LAAalyzer (detaillierter)	10
Abbildung 8 Funktionshierarchie LAAalyzer	10
Abbildung 9 Programmbausteine LAAalyzer	11
Abbildung 10 Programmablauf LAAalyzer	12
Abbildung 11 Programmfunktion WebFrontend	14
Abbildung 12 Funktionshierarchie WebFrontend	14
Abbildung 13 Programmbausteine WebFrontend	14
Abbildung 14 Programmablauf WebFrontend	15
Abbildung 15 Abhängigkeiten	19

Einleitung

Zusammen mit unserem Partner, Fa Inventek, haben wir dieses Tool entworfen, um dynamisch auf Log-Files unterschiedlichster Anwendungen zugreifen und diese nach definierbaren Regeln durchsuchen zu können, um hieraus dann z.B. in Monitoring-Werkzeugen Alarme zu erzeugen.

Funktionen

Ein/Mehrere periodisch geschriebene Log-Datei wird/werden gelesen, nach definierten Regeln ausgewertet bzw. analysiert und die Ergebnisse in einem Web-Frontend angezeigt. Weiterhin kann eine weitere Aktion, abhängig vom Ergebnis, ausgelöst werden, z.B. versenden einer E-Mail, schreiben eines Event-Logs (Ereignisanzeige) oder schreiben in eine Error-Log-Datei.

Eine Anbindung an TIVOLI wurde im Rahmen dieses Whitepapers zugrunde liegenden Projektes realisiert.

Systemübersicht

Das Gesamtsystem "LogAnalyzer" ist aus mehreren Subsystemen aufgebaut.

- Reader
- Analyzer
- Web-Frontend
- Datenbank

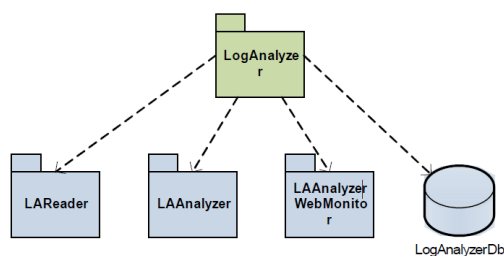


Abbildung 1 Gesamtsystem

Subsysteme

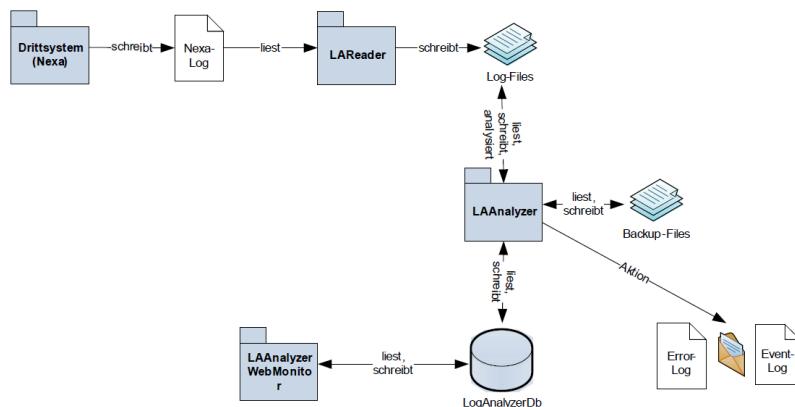


Abbildung 2 Subsysteme

Subsystem – Reader

Programmkurzbeschreibung

Der Reader synchronisiert die Log-Datei mit lokalen, vom Reader angelegten, Log-Dateien.

Programmfunktion

Der „Reader“ liest Einträge aus dem Log aus und schreibt diese in ein oder mehreren lokale(n) Log-Datei(en). Die Größe der Log-Dateien ist über eine Konfigurationsdatei einstellbar.

Es wird immer nur der neu hinzugekommene Eintrag (am Dateiende) im Log gelesen und synchronisiert.

Der Reader wurde als Konsolenprogramm wie auch als eine Dienstanwendung realisiert.

Datenorganisation/Voraussetzungen

Das Drittsystem ermöglicht oft genug den Zugriff auf das Log um Daten von dort zu lesen. D.h. es ermöglicht dauerhaft lesenden Zugriff, es öffnet die Datei nicht exklusiv. Nicht im Verantwortungsbereich des Readers liegt das Löschen der Log-Datei, dies muss das Drittsystem übernehmen. Das Löschen der lokalen Log-Dateien übernimmt der Analyser.

Funktionshierarchie

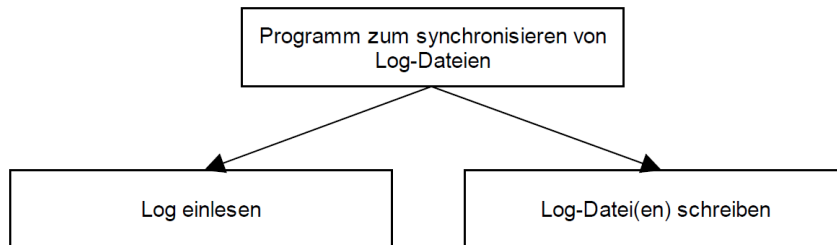


Abbildung 3 Funktionshierarchie LARReader

Fehlerbehandlung

Bei einem Fehler wird das Programm beendet.

Anwendung

Der Reader muss als Dienst installiert und auf „automatisch starten“ eingestellt werden. Der Reader läuft immer und verbraucht erst Prozessorleistung wenn die Datei geschrieben wird.

Programmaufbau

Programmbausteine

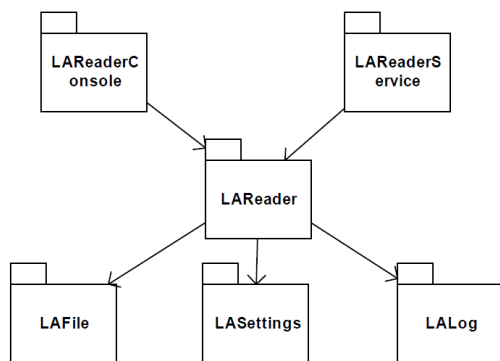


Abbildung 4 Programmbausteine LARReader

Programmablauf

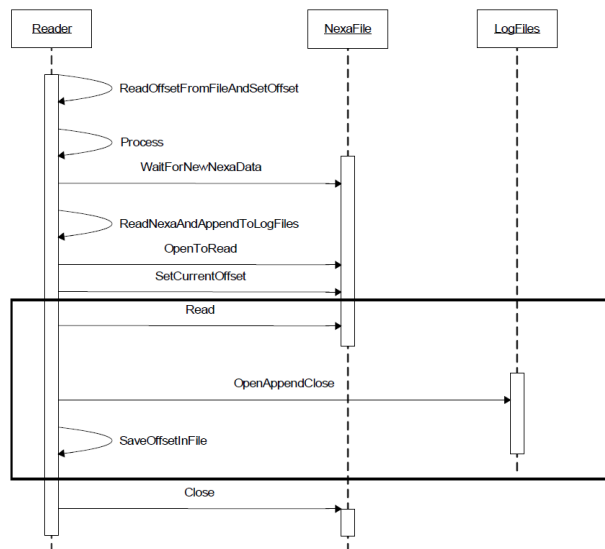


Abbildung 5 Programmablauf LAReader

Programmablaufbeschreibung

1. Der Synchronisierungsprozess startet in einem eigenen Thread.
2. Alter Offset-Wert aus *.ini – Datei lesen.
3. Es wird geprüft, ob das Log existiert, wenn nicht wird gewartet.
4. Verzeichnispfade und der Dateiname sind über eine Konfigurationsdatei einstellbar.
5. Wenn das Log existiert wird geprüft ob neue Daten zum synchronisieren vorhanden sind.
6. Sind keine Daten zum schreiben vorhanden, wird gewartet.
7. Sind neue Daten vorhanden, wird synchronisiert.
8. Datei wird nicht exklusiv geöffnet.
9. Datei wird gelesen.
10. Vorhandene lokale Logdatei- Namen einlesen.
11. Diese auf die Größe hin analysieren, eventuell neue Log- Datei erstellen + Nummerierung.

12. Log- Datei exklusiv öffnen.
13. Log- Datei schreiben.
14. Log-Datei schließen.
15. Aktuelle Offset der Datei in einer *.ini - Datei speichern. Notwendig für Feststellung neu hinzugekommener Daten im Log.
16. Punkte 8 – 15 werden solange wiederholt, bis Dateiende im Log erreicht.
17. Log wird geschlossen
18. Punkte 3 – 17 werden solange wiederholt, wie der Thread aktiv ist.

Subsystem - Analyzer

Programmkurzbeschreibung

Der Analyzer wertet Log- Einträge nach definierten Regeln aus.

Programmfunktion

Der Analyzer analysiert für jede Regel ("Rules") die Log-Datei(en) um zutreffende Einträge zu finden, die dann in einer Datenbank gespeichert werden. Die Regeln nach denen zu suchen ist, werden ebenso in einer Datenbank hinterlegt. Für jede Regel können zusätzliche Aktionen (E-Mail, Event-Log, Error-Log, ...) definiert werden, die nach Abschluss der Auswertung ausgeführt werden. Nach Abschluss der Analyse werden die analysierte(n) Log-Datei(en) in ein Backup-Verzeichnis verschoben.

Der Analyzer wurde als Konsolen- wie auch als Dienstanwendung realisiert.

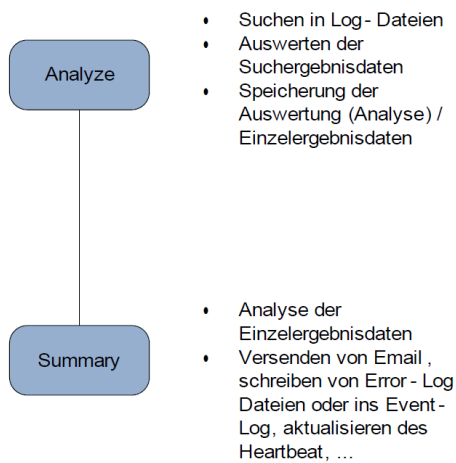


Abbildung 6 Programmfunktion LAAalyzer

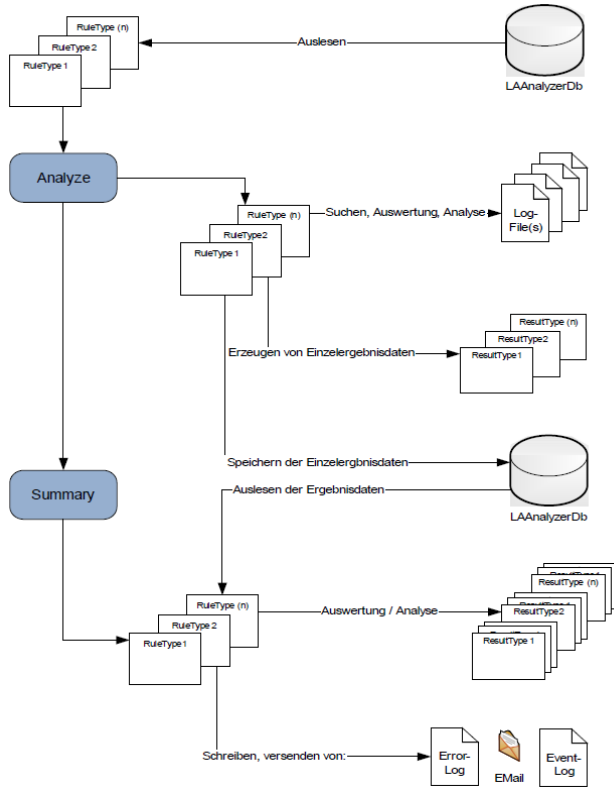


Abbildung 7 Programmfunktion LAAalyzer (detaillierter)

Datenorganisation/Voraussetzungen

Der Analyzer öffnet die lokalen Log-Dateien exklusiv. Nach der abgeschlossener Analyse der Log- Einträge werden Log- Dateien in ein Backup- Verzeichnis verschoben. Dieses Backupverzeichnis muss durch einen Benutzereingriff gelöscht werden.

Funktionshierarchie

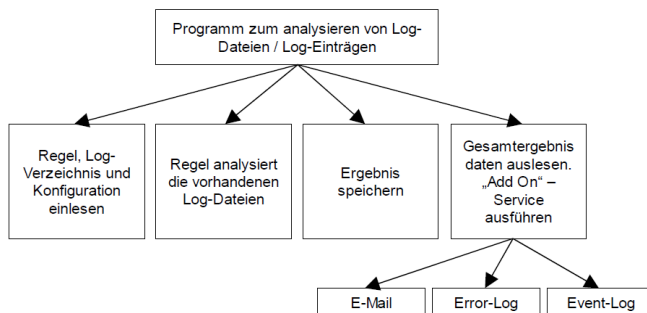


Abbildung 8 Funktionshierarchie LAAalyzer

Fehlerbehandlung

Bei einem Fehler wird das Programm beendet.

Anwendung

Der Analyzer muss als Dienst installiert und auf „automatisch starten“ eingestellt werden. Der Analyzer läuft immer und verbraucht erst Prozessorleistung wenn die Synchronisierung Datei => Log – Dateien beginnt.

Programmaufbau

Programmbausteine

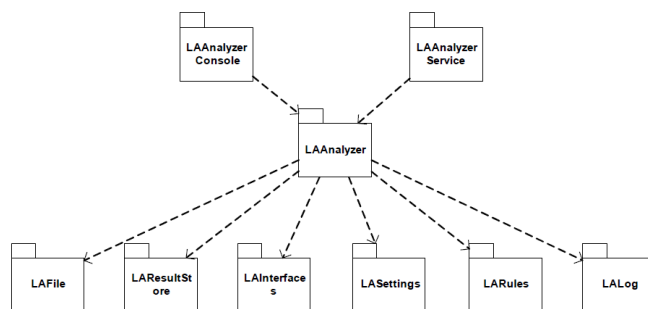


Abbildung 9 Programmbausteine LAAAnalyzer

Programmablauf

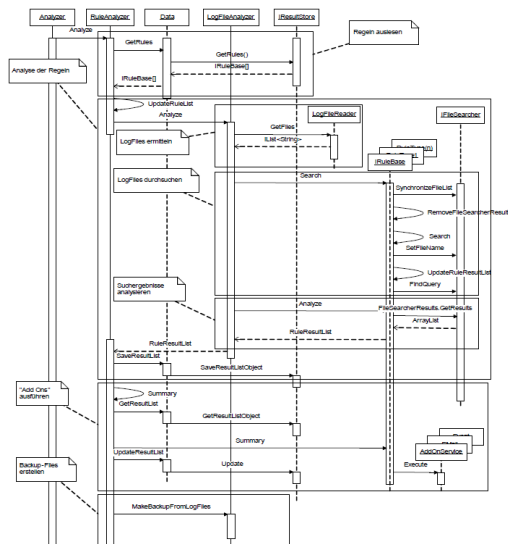


Abbildung 10 Programmablauf LAAnalyzer

Programmablaufbeschreibung

1. Der Analyse- Prozess startet in einem eigenen Thread.
2. Regel(n) aus der Datenbank auslesen.
3. Regel(n) ist definiert, dann
4. Log-Dateinamen einlesen.
5. Log-Dateien sind vorhanden, dann
6. Regel durchsucht Log-Dateien nach relevanten Log-Einträgen.
7. Regel analysiert Suchergebnis(se) und erzeugt Einzelergebnis(se).
8. Regel speichert Einzelergebnis(se).
9. Regel liest Gesamtergebnisse aus.
10. Regel analysiert Gesamtergebnisse und führt „Add Ons“ aus.
11. Für jede Regel werden Punkt 4 – 10 ausgeführt.
12. Backup- Dateien erzeugen und Log-Dateien löschen.
13. Punkte 2 – 12 werden solange wiederholt, wie der Thread aktiv ist.

Konfiguration

- LAAnaylzerSettings.xml
- LAEventLogSettings.xml
- LAHeartBeaterSettings.xml
- LAMailSettings.xml
- LAResultStoreDBSettings.xml
- LAResultStoreSettings.xml
- LARulesSettings.xml
- LASearcherSettings.xml

Siehe Kapitel „System - Konfiguration“.

Subsystem - WebFrontend

Programmkurzbeschreibung

Web-Frontend zur Anzeige der Einzelergebnisdaten und zur Konfiguration von Benutzern und Regeln.

Programmfunktion

Das web-basierenden Frontend bringt die Suchergebnisse d.h. die gefundenen relevanten Log Einträge zur Anzeige. Das Frontend bietet auch eine Möglichkeit, nicht mehr benötigte oder erfolgreich bearbeitete Einträge zu markieren bzw. zu löschen.

Für das Bestätigen von angezeigten Ereignissen (logisch "Löschen") muss sich ein Anwender eingeloggt haben, dafür ist eine einfache Benutzerverwaltung implementiert.

Die Ergebnisse der Auswertung werden in Listenform dargestellt.

Über das Frontend können, Benutzer und Regeln angelegt, geändert und gelöscht werden.

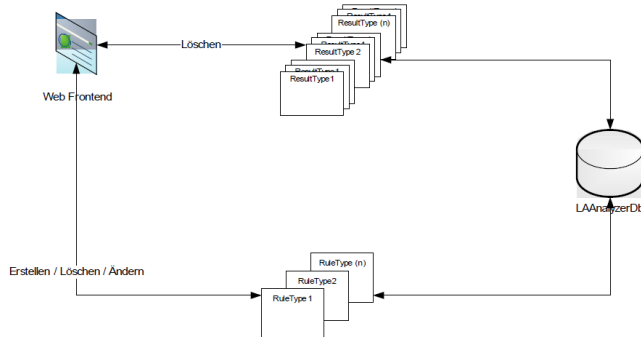


Abbildung 11 Programmfunktion WebFrontend

Funktionshierarchie

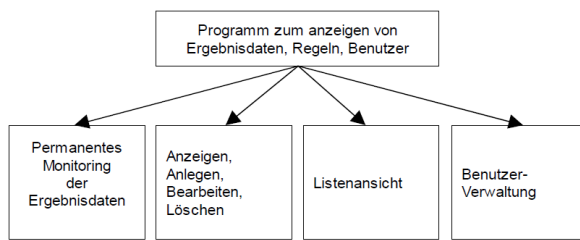


Abbildung 12 Funktionshierarchie WebFrontend

Fehlerbehandlung

Keine Fehlerbehandlung definiert. Fehler werden auf einer ASP – Seite angezeigt.

Programmaufbau

Programmbausteine

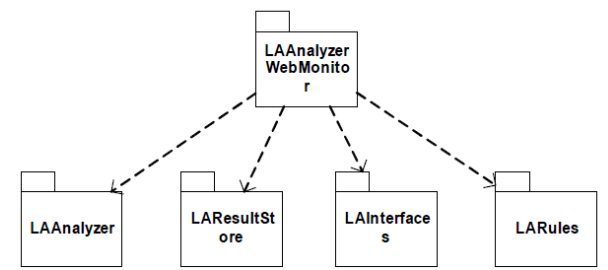


Abbildung 13 Programmbausteine WebFrontend

Programmablauf

Prinzip:

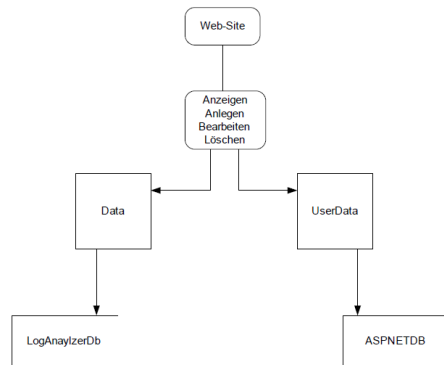


Abbildung 14 Programmablauf WebFrontend

Programmablaufbeschreibung

Prinzip:

1. Aktionen durch den Benutzer, wie Anzeigen der Einzelergebnisdaten, Anlegen von Regeln usw. ..., geschieht über die Zentrale Datenklasse „Data“.
2. Für die Benutzerverwaltung wird die Standard „ASPNETDB“- Datenbank verwendet. Alle Zugriffe werden Zentral über die Klasse „UserData“ verwaltet.

Konfiguration

Siehe Kapitel „System - Konfiguration“.

Über die Web- Config werden folgende Einstellungen vorgenommen:

Element	Beschreibung
PathToAppData	Der Pfad zu den Konfigurations- Dateien
UpdateTime	(1s => 1000), Aktualisierungszeit (Teilaktualisierung)
ScriptManagerAsyncPostBackTimeout	Asyn. Post-Back-Timout des Script-Managers

Subsystem - Datenbank

Die Datenbank speichert die Ergebnisdaten (relevante Log-Einträge) und die Regeln. Eine weitere Datenbank ist für die Benutzerverwaltung zuständig.

Datenbankname

Name: „LogAnalyzerDb“

Datenbanksystem

Datenbanksystem: SQL Server 2005 Express

Die Regeln wie auch dessen Ergebnissen werden als Objekte in XML- Format in den jeweiligen Tabellen gespeichert. Dadurch kann dieselbe Tabelle („tblRules“) für alle Regeln bzw. dieselbe Tabelle („tblResults“) für alle Ergebnisse verwendet werden.

Programminstallation

Softwarebedarf

- Internet Information Services (min. IIS 5.1).
- .NET Framework 2.0.
- Microsoft SQL Server Express.
- SQL Management Studio für SQL Server Express.
- Installations-Skript zur Datenbank „LogAnalyzerDb“.
- Das Tool „LOG-Analyser“.
- Setups der Subsysteme

Installationsanweisung

Bevor die Subsysteme des Log- Analyzers installiert werden können, müssen auf dem Zielsystem folgende Softwarekomponenten vorhanden sein.

- IIS (Mindestens IIS Version 5.1)
- .NET Framework 2.0 / Windows-Installer 3.1

Anmerkung zu 1. und 2.:

Wird der IIS nach dem .NET Framework 2.0 installiert, muss das .NET Framework 2.0 nochmal, nach der Installation des IIS, installiert werden.

- Microsoft SQL Server Express.
- SQL Management Studio für SQL Server Express.
- Serverauthentifizierung für SQL Server Express einrichten (Siehe Kapitel „Subsystem-Datenbank“)
- Die Datenbank „LogAnalyzerDb“ installieren. Hierzu folgende Skripts ausführen. (Achtung ! Reihenfolge ist entscheidend.)

Installation Web-Monitor

Das Setup „LAWebSetup.msi“ installiert den Web-Monitor auf das Zielsystem. Nach der Installation des Monitors sind folgende Schritte durchzuführen:

- Eine „Log-Source“ für die Ereignisanzeige unter Windows muss erstellt werden. Hierzu muss das Tool „LOG-Analyser“ verwendet werden.
- Lese- und Schreibrechte für alle erforderlichen Verzeichnissen und Datenbanken erteilen.

Installation Reader

Das Setup „LAReaderConsoleSetup.msi“ installiert der Reader als Konsolenanwendung auf dem Zielsystem.

Das Setup „LAReaderServiceSetup.msi“ installiert den Reader als Dienstanwendung auf dem Zielsystem.

Installation Analyzer

Das Setup „LAAalyzerConsoles.msi“ installiert den Analyzer als Konsolenanwendung auf dem Zielsystem.

Das Setup „LAAalyzerServices.msi“ installiert den Analyzer als Dienstanwendung auf dem Zielsystem.

Programmaufbau

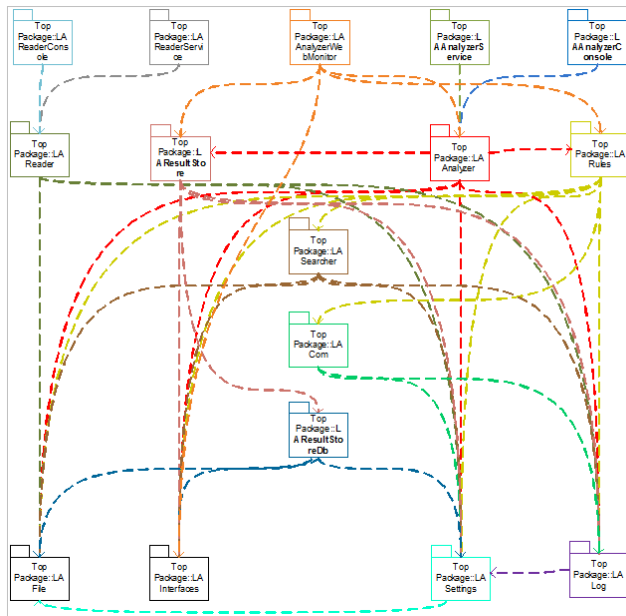


Abbildung 15 Abhängigkeiten