Dependency Index: Kennzahlen für ein großes DB- und Software-Refactoring

Thomas Rahn
LichtBlick SE
Hamburg

Schlüsselworte
Kennzahlen, Refactoring, Domäne, Dependency, Komplexität

Einleitung

LichtBlick steht vor einer Herausforderung: eine Vielzahl von Anwendungen sind über die Datenbank und Serverdienste mit einander verknotet. Wartung und Weiterentwicklung der Anwendungen wird zunehmend aufwendiger weil die Abhängigkeiten immer größer werden. Die Lösung ist die Trennung der Anwendungen und DB-Objekte in klar definierte Domäne mit Enterprise Service Bus für domänen-übergreifende Events und Datenfassaden für Datenanfragen aus anderen Domänen. Daraus entseht ein großes langjähriges Projekt mit vielen Teilprojekten. Um Fortschritt zu messen wird eine Dependency Index täglich aktualisiert, darin werden domäne-übergreifende Aufrufe gezählt, und gleichzeitig aufgezeigt wo der größte Handlungsbedarf besteht. Basis hierfür sind Oracle System Views und ein Scan von allem kompilierten Code. Die Ergebnisse werden in Oracle zusammengetragen, ausgewertet und von dort publiziert.

Wachsende Software Komplexität

LichtBlick ist mit über 600.000 Kunden der größte konzernunabhängige Anbieter von sauberer Energie. Mit innovativen Lösungen wie SchwarmEnergie und ZuhauseStrom ist LichtBlick heute die treibende Kraft auf dem Weg zu einer nachhaltigen Zukunft der Energie.

Ein wichtiger Beitrag zum LichtBlick Erfolg ist die konsequente Unterstützung aller Geschäftsprozesse durch IT-Systeme. Inzwischen sind es etwa 20 Hauptanwendungen und noch mal so viele unterstützende Systeme aus eigener Entwicklung. Zusätzlich gibt es etwa 10 gekaufte Systeme.

Die Kommunikation der Systeme untereinander war bisher unterschiedlich gelöst. Die ältere und noch vorherrschende Methode ist über mehrere Oracle Datenbanken. Alle Systeme greifen auf dieselben Daten. Über regelmäßige Abfragen bestimmter Parameter wissen nachgeschaltete Systeme von den Ereignissen die sie interessieren. Mit der Einführung der Client/Server Architektur auf .Net Basis vor fünf Jahren ist der Aufruf fremder Dienste inzwischen die zweit-häufigste Form der System-Kommunikation. Der Vorteil beider dieser Kommunikationsarten ist das sie relativ einfach zu implementieren ist und die bestehende Architektur nutzt.

Mit der Zeit zeigt sich der Nachteil dieser Systemkommunikation. Die zunehmende Komplexität der Verbindungen der Systeme zu einander und zu der Datenbank machen Wartung, Anpassungen und Neuentwicklung immer aufwendiger und langsamer (Abbildung 1).

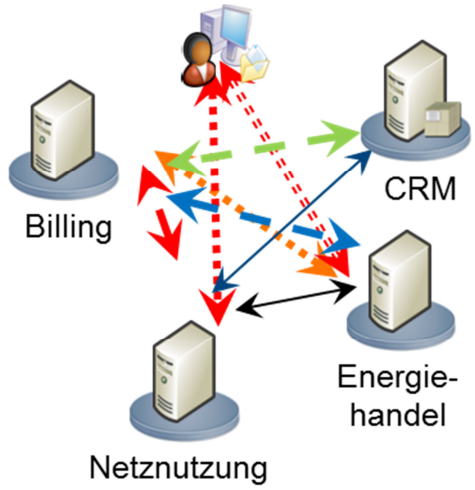

Abbildung 1: Bisherige Vernetzung der Systeme

Als Lösung haben wir die Einführung von Domänen und die Trennung aller Software Artefakten nach Domäne gewählt. Die Kommunikation zwischen den Domänen läuft dann nur noch über definierte Schnittstellen. Für die Kommunikation von Events haben wir uns für BizTalk entschieden, lesende Datenzugriffe sollen über Query Fassaden abgedeckt werden (Abbildung 2).

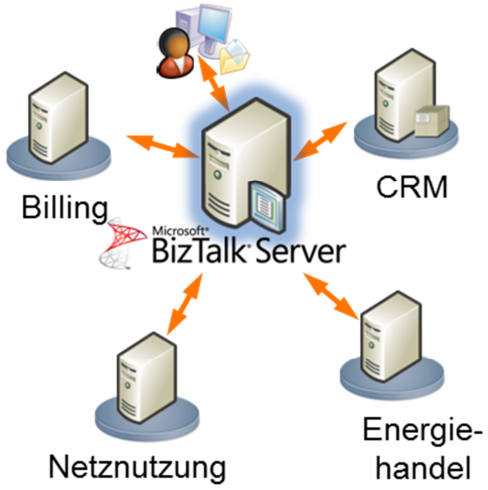

Abbildung 2: Neue Hub- und Spoke Architektur und Trennung der Domänen

Dependency Index

Mit der Zuordnung aller Software-Artefakte zu Domänen kamen Fragen zu den Abhängigkeiten auf.

- Welche Domänen sind von welchen anderen Domänen abhängig und wie stark?
- Wie können wir Änderungen in der Abhängigkeit messen?

- Wo müssen wir ansetzten um die Abhängigkeit zu reduzieren?

Um diese Fragen zu beantworten wurde der Dependency Index erfunden. Der Dependency Index soll regelmäßig und im Detail erfassen welche Software Artefakten domänen-übergreifend auf welche anderen Software Artefakten zugreift. Die einzelnen Zugriffe werden pro Domäne zum Dependency Index summiert.

Es gibt drei Arten von Abhängigkeit:

1. Anwendung zu Anwendung
2. Anwendung zu Datenbank
3. Datenbank zu Datenbank

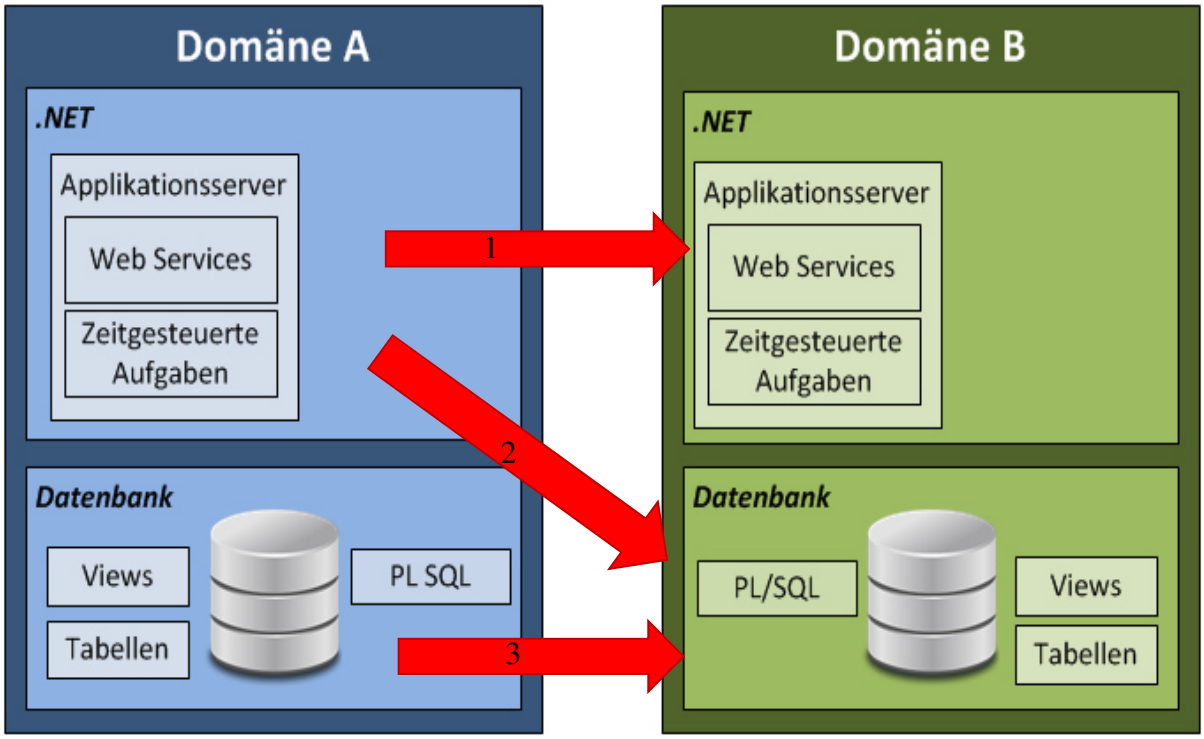

Abbildung 3: Arten von domänen-übergreifende Abhängigkeiten

Für die ersten zwei Arten der Abhängigkeit wird der produktive .Net Code dekompiliert und nach entfernten Aufrufen durchsucht. Gefundene Aufrufe werden in der Datenbank (Oracle) gespeichert. Domänen-übergreifende Datenbank zu Datenbank Aufrufe werden auf Basis der Oracle Dependency View ermittelt und ebenfalls in der Datenbank gespeichert. Die Daten werden über eine Serie von Views ausgewertet um folgende Ergebnisse pro Art der Abhängigkeit darzustellen:

- Aufrufendes Software-Artefakt und aufgerufenes Artefakt mit Domänenzuordnung
- Top aufrufende Software-Artefakte mit Anzahl Aufrufe und Domäne die am häufigsten aufgerufen wird.
- Top aufgerufene Software-Artefakte mit Anzahl Aufrufe und Domäne die am häufigsten das Artefakt aufruft.
- Zusammenfassung der Aufrufe pro Domäne zu anderen Domänen

Zusätzlich werden folgende zeitliche Verläufe der Abhängigkeiten pro Domäne dargestellt

- Anzahl Aufrufe fremder Domäne pro Domäne
- Anzahl Aufrufe durch fremde Domäne pro Domäne
- Summierung der Aufrufer und Aufgerufenen Artefakte pro Domäne

Der Dependency Index wird im LichtBlick Intranet und in einer Excel Datei (Management-kompatibel) dargestellt. Beide sind an die Oracle Auswertungs-Views gebunden.

Gestaltungsfragen beim Dependency Index

Die wichtigste Frage war was zum Dependency Index zählt. Diese Diskussion hatte mehrere Teilfragen. Wir haben wie folgt entschieden:

Jeder fremder Aufruf zweimal gezählt wird – einmal bei der Domäne die aufruft und einmal bei der Domäne die aufgerufen wird. Wir haben uns bewusst dazu entschieden weil die Entkopplung in den meisten Fällen von den Verantwortlichen beider Domäne umgesetzt wird.

Jeder Fund wird gleich bewertet. Wir hatten, z.B., überlegt das Trigger einen höheren Indexscore bekommen als Views, weil wir einige Trigger haben die als Eventmelder funktionieren und deshalb Handlungen andere Systeme auslösen – Views aber nur einen lesenden Zugriff darstellen. Auch ob ein großes PL/SQL Package, mit vielen Referenzen auf eine DB-Objekt, gleich behandelt werden soll wie eine Prozedur die ein Objekt nur einmal aufruft. Durch die unterschiedlichen Erhebungsmethoden zwischen den Abhängigkeitsarten entstehen weitere Ungleichheiten. Wir haben beschlossen dass der Index unnötig Komplex und nicht nachvollziehbar wird wenn wir versucht hätten alle diese Unterschiede zu berücksichtigen.

Es gibt Ausnahmen. Wir haben an verschiedenen Stellen die Möglichkeit für Ausnahmen eingebaut. Ein Grund waren False-Positves - die Such nach Referenzen auf ein Datenbankobjekt namens ‚X' macht wenig sinn. Dann gibt es aus verschiedenen Gründen mehrere Produktive Versionen des gleichen Software-Artefakts (manchmal auch mit kleinen Unterschieden). Diese werden nur einmal gezählt.

Wie der Dependency Index den Umbau der Software unterstütz

Weil die Domänen nach Teams geschnitten sind, finden sich die Teams in den Ergebnissen wieder. Sie können zeitnah (am nächsten Tag) das Ergebnis Ihrer Entkopplungsarbeit nachvollziehen. Gute Arbeit wird bestätigt.

Der Dependency Index funktioniert als Macro Projekt-Management-Tool. Die Umsetzung des Projekts erfolgt über sehr viele kleine und große Teilprojekte, mit unterschiedlichen Teilnehmern. Eine zentrale Planung und Steuerung dieser Projekte ist nur zum Teil gewünscht und möglich. Jedes Team soll weitestgehend frei entscheiden wann und wie sie zum Erfolg des Gesamt-Projekts beiträgt. Dennoch ist es nötig Ziele zu setzen. Der Index ermöglicht es Ziele zu definieren – z.B. Senkung des Indexes um 25% bis Ende 2014 - und den Vollzug zu kontrollieren.

Auch als Werkzeug für die detaillierte Planung von Teilprojekten lässt sich der Index einsetzen. Die einzelnen Auswertungs-Views geben klare Aussagen über die Software-Artefakte die geändert werden müssen. Im Einzelnen kann der Index auf folgender Weise reduziert werden:

1. Falsche positive Dependencies melden.
2. In der falschen Domäne zugeordnete Software-Artefakte ändern.
3. Dependencies ablösen - Kommunikation zwischen den Domänen über Fassaden und BizTalk laufen lassen.

4. Nicht genutzten Code und DB-Objekte löschen.

Der Index ist öffentlich. Die Teams sehen nicht nur ihren eigenen Erfolg, sondern auch den der anderen Teams. Der Index fördert damit die Konkurrenz zwischen den Teams – als Beispiel, in Abbildung 4, Teil einer Email nachdem die Abhängigkeit einer Domäne unter den einer anderen Domäne gesunken war und die Kollegen sich darüber freuten.

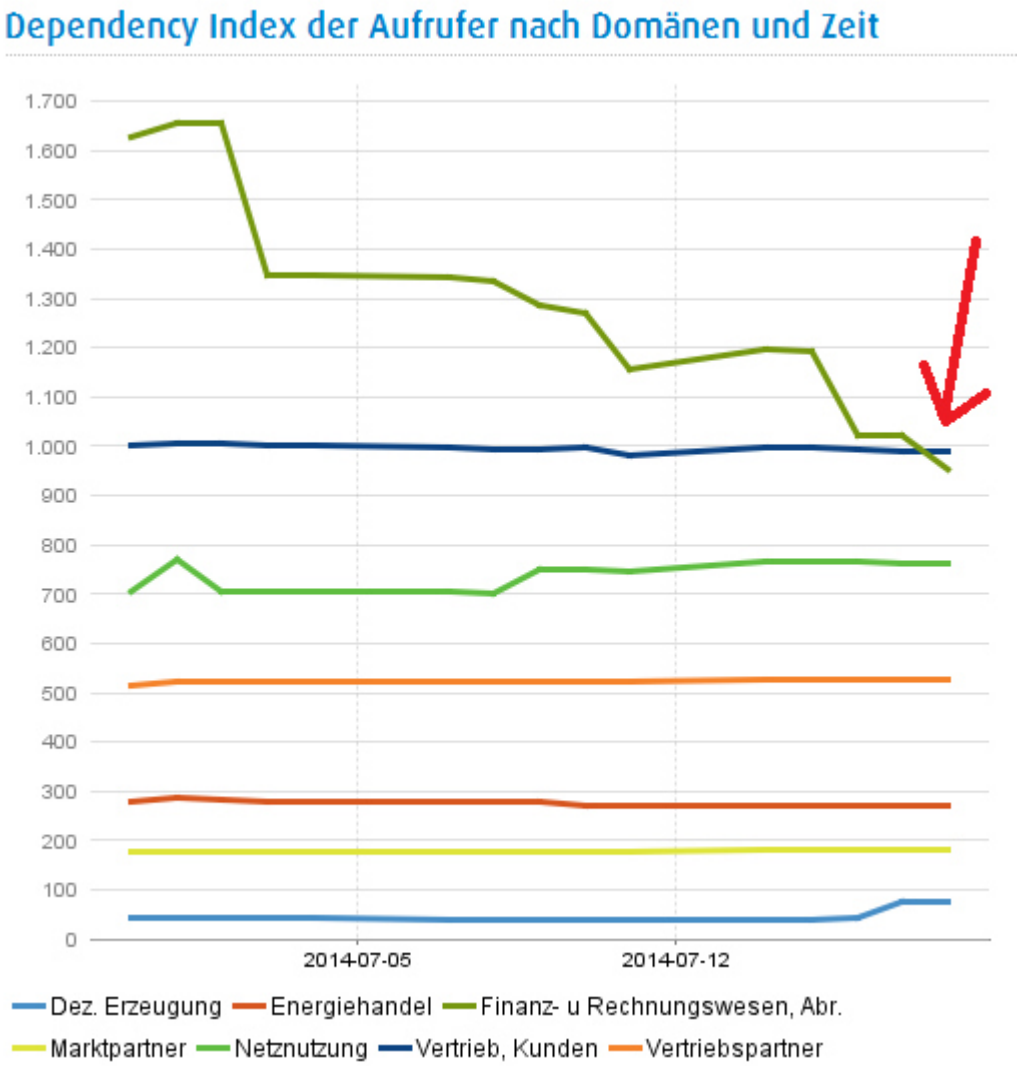

Abbildung 4: Teams konkurrieren um Ergebnisse im Dependency Index

Fazit

Für eine wichtige Softwarearchitektur-Änderung, die eine Vielzahl von Systemen betrifft und nur über einen mehrjährigen Zeitraum umgesetzt werden kann, sind Kennzahlen über Projekt-Fortschritt ein wichtiges Werkzeug für die Planung, Kontrolle und Kommunikation.

Kontaktadresse:

Thomas Rahn

LichtBlick SE

Zirkusweg 6

D-20359 Hamburg

Telefon:	+49 (0) 40-6360-1384
Fax:	+49 (0) 40-6360-2130
E-Mail	thomas.rahn@ lichtblick.de
Internet:	www.lichtblick.de