Oliver Linssen et al. (Hrsg.): Projektmanagement und Vorgehensmodelle 2019,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2019

Open Digital Assistant Framework – ein quelloffenes Framework für projektbasierte digitale Assistenten

Olaf Resch[1], Aglika Yankova[2]

Abstract: Das Open Digital Assistant Framework soll die Realisierung projektbasierter digitaler Assistenten durch die Bereitstellung zentraler Grundfunktionalitäten vereinfachen. Dem Projektmanagement kommt bei digitalen Assistenten einerseits eine Enabler-Funktion zu, da es eine zielorientierte und mehrere Schritte umfassende Struktur zur Bearbeitung komplexer Aufgaben liefert. Andererseits können digitale Assistenten das Projektmanagement durch eine stetige Informationsversorgung der Beteiligten und durch die kontinuierliche Erfassung wichtiger Projektdaten wie beispielsweise Fertigstellungsgrade bereichern. Zwar existiert das Open Digital Assistant Framework selber aktuell nur als früher Entwurf, jedoch gibt es mit dem Open Knowledge Interface einen digitalen Assistenten in einem fortgeschrittenen Entwicklungsstadium, der als Grundlage für das Framework dienen kann. Der vorliegende Kurzbeitrag beschreibt zunächst das Open Knowledge Interface als konkreten digitalen Assistenten und entwickelt darauf basierend die Idee des Open Digital Assistant Framework als universellem Framework zur Realisierung projektbasierter digitaler Assistenten.

Keywords: Digitaler Assistent, Projektmanagement, Open Source, Framework

1 Einleitung

Digitale Assistenten unterstützen ihre Anwender bei der Bearbeitung komplexer Aufgaben. Beispiele für solche komplexen Aufgaben sind das Verfassen einer wissenschaftlichen Hausarbeit, die Entwicklung einer Software oder die Geldanlage zur Vermögensbildung. Komplexe Aufgaben können durch Zerlegung in Teilaspekte wie Vorgänge, Stakeholder und Termine vereinfacht und strukturiert werden. Diese Zerlegung erleichtert es dem menschlichen Bearbeiter zum richtigen Zeitpunkt das Richtige zu tun; für den Einsatz eines digitalen – maschinellen – Bearbeiters, ist eine exakte Struktur jedoch eine absolute Grundvoraussetzung. So kann der besonders begabte Student seine Bachelorthesis eventuell auch in einer kreativ unstrukturierten Weise mit einer hohen Ergebnisqualität fertigstellen. Der digitale Assistent benötigt jedoch immer eine detaillierte Struktur und das Projektmanagement liefert den Rahmen dafür. Solch eine Struktur bedeutet natürlich nicht, dass ein Assistent nicht flexibel reagieren kann. Es muss nur klar definiert sein, worauf sich diese Flexibilität bezieht. Beispielsweise können und werden sich einmal geplante Termine im Laufe des Projektes ändern. Damit kann auch der digitale Assistent umgehen. Er hätte allerdings Schwierigkeiten, wenn ein Termin auf einmal keine Zeitpunktgröße mehr ist, sondern eine Zeitintervallgröße. Anders ausgedrückt, stellt das Projektmanagement eine gemeinsame Sprache bereit, in welcher der digitale Assistent mit seinem menschlichen Anwender kommunizieren kann. Gleichzeitig eröffnen digitale

[1] Hochschule für Wirtschaft und Recht Berlin, Alt-Friedrichsfelde 60, 10315 Berlin, Olaf.Resch@hwr-berlin.de
[2] Hochschule für Wirtschaft und Recht Berlin, Alt-Friedrichsfelde 60, 10315 Berlin, Aglika.Yankova-Hristova@hwr-berlin.de

Assistenten auch neue Möglichkeiten für das Projektmanagement, z.B. die kontinuierliche Erfassung von Daten aus denen Aussagen über den Projekterfolg abgeleitet werden können.

Der vorliegende Beitrag stellt mit dem Open Knowledge Interface (OKI) zunächst einen konkreten digitalen Assistenten vor, der sich bereits im Einsatz befindet. Im Anschluss daran werden die Erkenntnisse aus dem OKI-Projekt verallgemeinert und die Idee einer universellen Softwarebasis zur Erstellung projektbasierter digitaler Assistenten eingeführt. Das vorliegende Ideenpapier verzichtet auf eine ausführliche Darstellung des Standes von Wissenschaft und Technik. Für eine gut verständliche Einführung in das Thema: digitale Assistenten siehe [ASW18]. Es existieren bereits Frameworks, die sich in den Bereich der digitalen Assistenten einordnen lassen, siehe z.B. [Bo19],[Mi19]. Diese unterstützen allerdings vorwiegend die Reaktion auf Benutzeranfragen als Chatbots. Das hier vorgestellte ODAF Framework geht darüber hinaus und ermöglicht die Erstellung von digitalen Assistenten zur Unterstützung komplexer Aufgaben, die ihre Nutzer zielorientiert über einen längeren Zeitraum unterstützen und dazu auch aktiv Dialoge initiieren.

2 Open Knowledge Interface

OKI unterstützt als digitaler Assistent Studierende beim Verfassen wissenschaftlicher Hausarbeiten. Das Forschungsprojekt ist an der Hochschule für Wirtschaft und Recht Berlin angesiedelt und läuft von Mai 2018 bis zum Oktober 2019. Unser besonderer Dank gilt dem Bundesministerium für Bildung und Forschung, das OKI in der Leitlinie Open Access unterstützt [Re18].

2.1 Technische Umgebung

Das OKI Frontend ist vorwiegend als Telegram Chatbot realisiert, siehe Abbildung 1 für einen Eindruck. Telegram stellt eine umfangreiche API zur Integration eigener Anwendungen bereit [Te19]. Das Aufrufen von Funktionalitäten, wie das Anlegen eines neuen Projektes, erfolgt durch die Telegram Befehlssyntax, z.B. /newproject oder durch eine natürlich-sprachliche Eingabe, z.B. „Lege ein neues Projekt an“. Für bestimmte Funktionalitäten ist der Chat allerdings eine suboptimale Benutzerschnittstelle, weil sie eine größere Ansicht erfordern, wie z.B. die Terminjustierung. Für diese Fälle nutzt OKI Java-Script-basierte Web-Seiten, die aus dem Chat aufgerufen werden.

Abb. 1 OKI als Telegram Chatbot

Der OKI Server ist vorwiegend in Node.js programmiert. Die Datenhaltung erfolgt in einer MongoDB Datenbank. Die einzelnen Funktionalitäten, z.B. das Anlegen eines Projektes, werden in einer eigens entwickelten Skriptsprache programmiert, Gleiches gilt für die einzelnen Projekttemplates, für z.B. eine Bachelorthesis oder einen Essay. Der Server enthält eine Logik zur Interpretation der Skripte. Funktionalitäten können sowohl reaktiv nach einer Benutzeranfrage als auch aktiv, z.B. zeitgesteuert aufgerufen werden. Die Interpretation natürlich-sprachlicher Eingaben erfolgt unter Verwendung von Rasa [Ra19]. Im Gegensatz zu API-basierten Lösungen läuft Rasa völlig autark auf der eigenen Hardware. Das ermöglicht eine wesentlich stärkere Kontrolle über das erzeugte Sprachmodell, erfordert aber auch eine aufwändigere Infrastruktur. Der Stack zur Realisierung der Spracherkennung umfasst deshalb neben Rasa auch noch Scikit-Learn [Sc19] und Spacy [Sp19]. Dieser Stack kann neben der Spracherkennung auch zur Umsetzung weiterer Aufgaben für das maschinelle Lernen verwendet werden.

2.2 Wissenschaftliche Hausarbeiten als Projekte

Wissenschaftliche Hausarbeiten werden in Form von Projekttemplates beschrieben. Ein Projekttemplate besteht aus:

- Ressourcen, z.B. Studierender und Betreuer.
- Vorgängen, z.B. Themenfindung und Literaturrecherche.
- Zeitansätzen für die Vorgänge, z.B. 16 Stunden für die Themenfindung.
- Eventuelle notwendige Vorgänger, z.B. Gewinnung Interviewpartner vor Interviewdurchführung.

- Eventuelle Wartezeiten, z.B. 10 Tage Wartezeit bei der Gewinnung von Interviewpartnern.
- Eventuelle Unterstützungsfunktionen für einzelne Vorgänge, z.B. automatische Literaturrecherche beim entsprechenden Vorgang.
- Eventuelle Hilfetexte für einzelne Vorgänge, z.B. Informationen zur Durchführung eines qualitativen Interviews beim entsprechenden Vorgang.
- Eventuelle fixe Termine, z.B. Besprechungstermine an der Hochschule.

Je nach Art der Arbeit, z.B. Bachelorthesis im Maschinenbau oder Essay in Wirtschaftskommunikation unterscheiden sich die Projekttemplates natürlich erheblich, sie setzen sich aber immer aus den gleichen Grundbausteinen zusammen. In technischer Hinsicht sind die Projekttemplates als JSON-Dateien abgebildet und somit maschinenlesbar. Prinzipiell können die JSON-Dateien auch durch Menschen editiert werden. Da dies jedoch eine hohe Technikaffinität voraussetzt, existiert in OKI eine eigene Anwendung zur einfachen Bearbeitung von Projekttemplates, prinzipiell durch jeden, der wissenschaftliche Arbeiten betreut.

Zum Anlegen eines Projektes muss der Studierende dann nur noch einen Namen vergeben, ein Projekttemplate auswählen und ein Startdatum festlegen. OKI kann daraufhin Vorgänge terminieren und den Studenten zu den entsprechenden Terminen an Aufgaben erinnern, mit relevantem Wissen versorgen und mit automatischen Funktionen unterstützen.

2.3 Zentrale Funktionalitäten eines projektbasierten digitalen Assistenten

OKI erstellt terminierte Aufgaben anhand vorgefertigter Templates. Die resultierenden Projektpläne können als Gesamtschau des Projektes durch den Nutzer reflektiert und eventuell angepasst werden. Eine zentrale Funktion ist daher die Anpassung der automatisiert erstellten Projektpläne. Speziell Studierende sollen Pläne nicht lediglich abarbeiten, sondern sie sollen diese verstehen, kritisch hinterfragen und anpassen. Die Aufforderung zur Planänderung erfolgt daher am Beginn aller Projekttemplates.

Eine vergleichsweise einfache Funktion ist das Erinnern an anstehende Aufgaben. OKI nutzt dazu den Telegram Chat und sendet eine entsprechende Nachricht. Die Nachricht kann zusätzliche Informationen enthalten: als Text, Sprachnachricht, Video oder als Verlinkung von Informationen, die an anderer Stelle gespeichert sind. OKI kann so seinen Nutzer je nach Projektfortschritt mit dem gerade relevanten Wissen versorgen.

Bestimmte Aufgaben können direkt durch OKI unterstützt werden. Die entsprechende Unterstützung erfolgt wieder genau dann, wenn diese Aufgabe ansteht. Beispielsweise ist die Literaturrecherche eine Aufgabe aller wissenschaftlichen Projekte. OKI unterstützt seinen Nutzer durch eine mobil optimierte Recherche und weist ihn regelmäßig auf aktuelle Publikationen hin.

OKI nimmt eine aktive Rolle im Dialog mit seinem Nutzer ein. Das bedingt allerdings auch, dass es sein kann, dass der Nutzer zum Zeitpunkt der Dialogaufnahme nicht verfügbar ist. Eine wichtige Funktion ist daher die Verschiebung von Terminen, entweder als Reaktion auf einen expliziten Benutzerwunsch oder weil der Nutzer auf eine Anfrage nicht reagiert. Ein Beispiel dafür ist die Aufforderung von OKI Suchbegriffe festzulegen. OKI benötigt diese Suchbegriffe für die automatisierte Literaturrecherche und muss daher den

Termin der entsprechenden Aufgabe so lange verschieben, bis der Nutzer die Suchbegriffe eingegeben hat.

Der digitale Assistent kann den Nutzer nach seinem Arbeitseinsatz und zum erfolgreichen Abschluss von Aufgaben befragen. Anhand dieser Daten können Plan und Ist abgeglichen und darauf entsprechend regiert werden, z.B. mit einer automatischen Plananpassung. Weitere Daten ergeben sich aus der Nutzung selber, z.B. kann OKI die Anzahl der durch den Studenten durchgeschauten Literaturabstracts speichern. Anhand dieser Daten kann OKI abschätzen, ob der Erfolg des wissenschaftlichen Projektes in Gefahr ist und den Studierenden entsprechend informieren. Dazu können sowohl regelbasierte Verfahren als auch Verfahren des maschinellen Lernens zum Einsatz kommen.

3 Open Digital Assistant Framework

OKI ist ein digitaler Assistent mit dem engen fachlichen Fokus wissenschaftlicher Hausarbeiten. Viele der zuvor skizzierten Funktionen lassen sich aber auch für ganz andere Projekte verwenden. Die enge Zusammenarbeit von digitalem Assistent und menschlichem Nutzer ermöglicht es, diesen stetig mit Wissen zu versorgen und erzeugt gleichzeitig Daten für eine kontinuierliche und automatische Adaption des Projektes an die Realität. Da dies für recht viele Projekte interessant ist, scheint es vielversprechend, bestimmte Funktionalitäten in ein Open Digital Assistant Framework (ODAF) auszugliedern.

Das ODAF kann als Open Source Projekt durch eine größere Gemeinschaft vorangetrieben werden, die alle identische Grundfunktionalitäten nutzen, die dadurch gekennzeichnet sind, dass sie zwar zwingend benötigt werden, um einen digitalen Assistenten bereitzustellen, die Funktionalitäten aber nicht unbedingt zur Differenzierung im Wettbewerb geeignet sind, z.B. Dialogführung und Projektplanung. Die Differenzierung erfolgt dann erst beim konkreten Einsatz des ODAF, durch Hinzufügen eigenständiger und wertvoller Kernfunktionalitäten, die normalerweise nicht quelloffen sind. Beispielsweise könnte ein digitaler Assistent einer Bank Kunden bei der Vermögensbildung unterstützen. Dieser Assistent arbeitet genau wie OKI projektbasiert und muss genauso wie OKI Dialoge führen, Termine verwalten, usw. – er würde diesen Grundfunktionalitäten aber spezielle Investmentfunktionalitäten hinzufügen, die einen originären Kundennutzen ermöglichen.

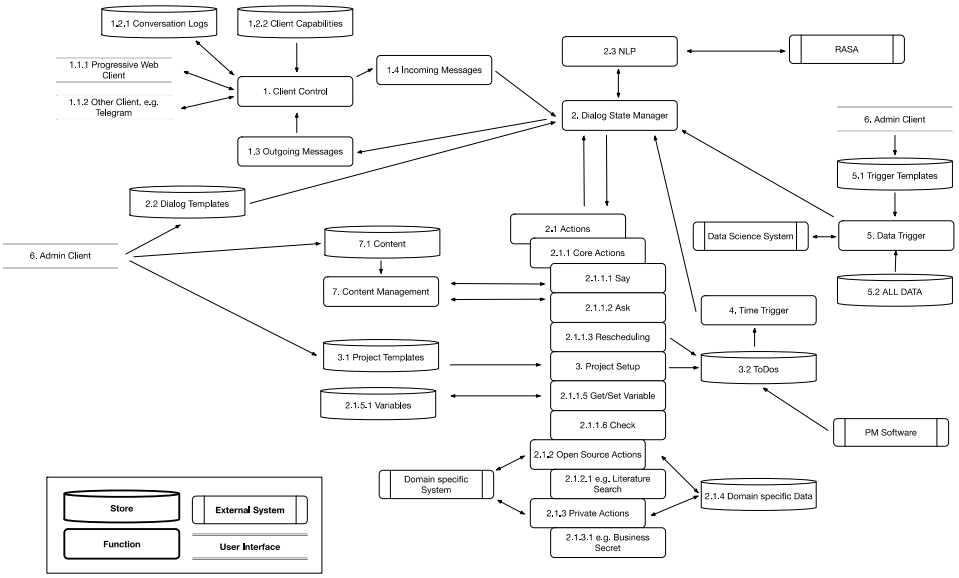

Abb 2: ODAF Übersicht [Re19]

Abbildung 2 zeigt eine einfache schematische Darstellung des ODAF. Ein Digitaler Assistent kann mithilfe verschiedener Clientumgebungen mit seinen Nutzern kommunizieren. Diese Clientumgebungen unter Beachtung der jeweiligen Möglichkeiten zu steuern und zu koordinieren ist Aufgabe der zentralen Funktion *1. Client Control*. Eine weitere zentrale Funktion ist der *2. Dialog State Manager*, der für die Dialogsteuerung zuständig ist. Eine Hilfsfunktion für die Dialogsteuerung ist die Verarbeitung natürlicher Sprache (Natural Language Processing). Diese Funktionalität muss allerdings nicht durch das ODAF abgedeckt werden, sondern kann von einer Fremdanwendung, in diesem Falle Rasa, bezogen werden.

Im Zuge eines Dialoges werden Aktionen ausgelöst, die der Dialogführung selber dienen, z.B. das Versenden von Nachrichten oder der jeweiligen Domäne zuzuordnen sind, z.B. die Literatursuche bei OKI. Domänenspezifische Aktionen können Schnittstellen zu entsprechenden domänenspezifischen Systemen aufweisen und sie können domänenspezifische Daten erzeugen. Wie bereits beschrieben, kann der Nutzer des ODAF selber entscheiden, ob er solche Aktionen der Gemeinschaft als Open Source beisteuern oder diese lieber für sich selber behalten will. Eine zentrale *Aktion ist 3. Project Setup,* das Anlegen von Projekten und die damit verbundene Erzeugung der ToDos, welche die Grundlage für die zeitgesteuerte Auslösung von Dialogen bilden. Neben der Erzeugung der ToDos auf Basis von Templates, können diese auch aus einer Projektmanagementsoftware importiert werden, was insbesondere bei sehr spezifischen Projekten sinnvoll ist, für die es nicht effizient wäre, ein eigenes Template anzulegen.

Dialoge können zeitgesteuert, durch *4. Time Trigger*, welche die ToDos auswerten, durch eine explizite Benutzeranfrage oder durch Datentrigger (*5. Data Trigger)* aufgrund einer bestimmten Datenkonstellation ausgelöst werden. Solche Datentrigger können auf alle verfügbaren Daten und auf externe Data Science Anwendungen zurückgreifen.

Dialoge beruhen auf Templates, die über eine separate Administrationsoberfläche (*6. Admin Interface)* gewartet werden. Diese Administrationsoberfläche dient auch zur Gestaltung von Templates für Projekte und Datentrigger sowie für die Wartung von Inhalten, z.B. Hinweistexten. Inhalte spielen für die Dialogführung eine erfolgskritische Rolle, weshalb es mit *7. Content Management* eine eigene zentrale Funktion für das Inhaltsmanagement gibt.

Das ODAF kann in eine gegebene Anwendungslandschaft integriert und je nach Einsatzzweck verwendet werden. Beispielsweise könnte ein einfacher reaktiver Chatbot alleine durch die Definition von Dialogtemplates erstellt werden. Für einen komplexen zeitgesteuerten digitalen Assistenten wie OKI müssen zusätzlich Projekttemplates gestaltet sowie domänenspezifische Aktionen programmiert werden. Soll der Assistent auch auf bestimmte Datenkonstellationen reagieren, wie z.B. eine Häufung nicht rechtzeitig abgeschlossener Aufgaben, sind außerdem Templates für Datentrigger notwendig. Wird der Assistent zur Unterstützung einzelner Projekte eingesetzt, muss die Schnittstelle zu der entsprechenden Projektmanagementsoftware gesteuert werden. Diese erzeugt einerseits die ToDos und kann andererseits als domänenspezifisches System durch domänenspezifische Aktionen angesprochen werden, beispielsweise um Plan- oder Ist-daten zu erfassen.

Im Gegensatz zu OKI existiert das ODAF bisher lediglich als frühes Konzept. Allerdings ist OKI bereits sehr generisch aufgebaut, weil die Templates für Projekte und Dialoge nicht hart-codiert sind, sondern in Form von JSON-Dateien vorliegen und zur Laufzeit interpretiert werden. Daher kann das Systemdesign für das ODAF auf die Vorarbeiten von OKI zurückgreifen. Der Realisierungsaufwand sollte dennoch nicht unterschätzt werden, da prinzipiell alles neu programmiert werden muss, um den Ansprüchen an eine offene Codebasis zu genügen. Ein weiterer erheblicher Aufwandsblock stellt die Neuentwicklung einer eigenen Clientumgebung dar (*1.1.1 Progressive Web Client*), die alternativ zu Telegram, bzw. anderen Fremdumgebungen eingesetzt werden kann. Es ist jedoch wichtig, diesen Aufwand zu betreiben, weil viele professionelle Szenarien bestimmte Anforderungen an die Benutzeroberfläche stellen, die durch die gängigen Fremdanbieter aktuell nicht erfüllt werden, wie beispielsweise eine anonyme Zugangsmöglichkeit, der Verzicht auf Zugriffslimits oder umfangreichere Möglichkeiten bei der Oberflächengestaltung.

Ein Open Access Projekt ermöglicht, dass der skizzierte Aufwand durch eine größere Gemeinschaft getragen wird und einer größeren Gruppe zugute kommt. Jeder kann dann selber entscheiden, ob er den quelloffenen Code ergänzen und das Projekt ODAF damit voranbringen will oder das ODAF lediglich nutzt und um selbst erstellte Funktionen erweitert, um damit einen Wettbewerbsvorteil zu erzielen.

4 Fazit

Der vorliegende Beitrag hat anhand des OKI-Beispiels gezeigt, wie ein digitaler Assistent Projektmanagement als Grundlage für eine gezielte Nutzerunterstützung einsetzt und wie digitale Assistenten die Möglichkeiten des Projektmanagements erweitern können. Da dies für recht viele Projekte neue Möglichkeiten eröffnet, wurde das ODAF als Konzept eines quelloffenen Frameworks für digitale Assistenten vorgestellt. Mithilfe des ODAF können digitale Assistenten schneller und aufwandsärmer entwickelt werden ohne auf die

Differenzierung eigener Assistenten verzichten zu müssen. Die Idee für das ODAF entstammt einem Workshop der Chatbot Community in Berlin, bei dem OKI vorgestellt wurde und bei dem vor allem die Technologie zur Steuerung aktiver Dialoge auf großes Interesse stieß. Allerdings kann ein Vorhaben wie das ODAF nur dann Erfolg haben, wenn es mittelfristig von einer größeren Gemeinschaft getragen wird.

Literaturverzeichnis

[ASW18] Apt, W.; Schubert, M; Wischmann, S.: Digitale Assistenzsysteme Perspektiven und Herausforderungen für den Einsatz in Industrie und Dienstleistungen, https://www.iit-berlin.de/de/publikationen/digitale-assistenzsysteme, Stand: 10.06.2019.

[Bo19] Bot UI: A Java Script Framework to build conversational UIs. https://botui.org/, Stand: 10.06.2019.

[Mi19] Microsoft Bot Framework: A comprehensive framework for building enterprise-grade conversational AI experiences. https://dev.botframework.com/, Stand: 10.06.2019.

[Ra19] Rasa Technologies Inc.: Rasa Stack: Open source conversational AI. https://rasa.com/products/rasa-stack, Stand: 10.06.2019.

[Re18] Resch, O.: OKI Open Knowledge Interface. http://oki2019.de, Stand: 12.07.2019.

[Re19] Resch, O.: Open Digital Assistant Framework High Level Model Sketch. https://github.com/ajdfnwjf/odaf/blob/master/README.md, Stand: 12.07.2019.

[Sc19] Scikit-Learn: Machine Learning in Python. https://scikit-learn.org, Stand: 10.06.2019.

[Sp19] Spacy: Spacy: Industrial-Strength Natural Language Processing. https://spacy.io/, Stand: 10.06.2019.

[Te19] Telegram: Telegram Bot API. https://core.telegram.org/bots, Stand: 10.06.2019.