ARCHITECTURE OF READING-COMPREHENSION ASSISTIVE TOOL – OPEN BOOK

Nikolay Pavlov

Abstract. This paper describes the architecture of Reading-Comprehension Assistive Tool – Open Book. It starts with the primary considerations when designing the architecture. The paper advises a four-tiered architecture and presents the advantages of the chosen solution. Finally, the paper presents the components of the tool and how they interact with each other.

Keywords: Open Book, architecture, distributed system

1. Introduction

This paper describes the architecture of Open Book [1], a distributed software system employing various natural language processing components to convert documents for people with (Autistic Spectrum Disorders) ASD in a format easier to read and understand. Open Book is developed as a part of project FIRST (A Flexible Interactive Reading Support Tool) [2] partially funded by the European Commission under the Seventh Framework Programme (FP7-2007-2013) for Research and Technological Development under grant agreement № 287607.

The FIRST project aims to improve the life of people with ASD by enabling them to read written documents and thus improving their social inclusion and interaction. People with ASD may experience difficulties comprehending complex instructions while reading, getting misled by figurative language or the use of rare words or, simply, get distracted by secondary points touched upon in a document [3]. Using innovative language technology to simplify documents, the created software product Open Book helps ASD users convert a standard document into a personalized version which is easier for them to read and understand.

With Open Book, users can convert documents they wish to access into a personalized form that facilitates reading comprehension. The conversion process involves automatic detection of linguistic features in the input document that are likely to impede comprehension and removal of those obstacles while preserving as much of the original meaning of the document as much as possible. The figure below presents an overview of the proposed solution and how different types of users interact with the software.

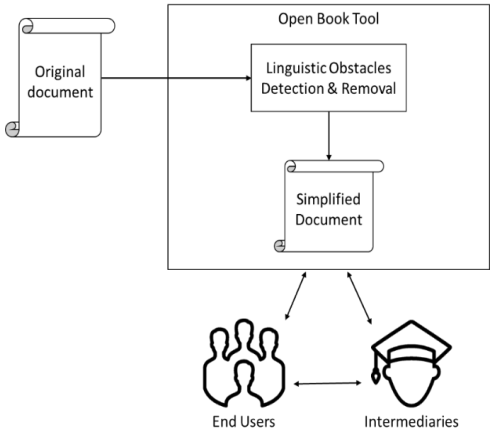

Figure 1. Open Book Tool

2. Considerations

In order to decide the most optimal architectural style/pattern that would be best suited for the Open Book Solution, the following factors were considered:

- Ability to re-use code across disparate environments so that business specific logic can be shared allowing possibilities towards infinite extendibility.
- Ability to horizontally scale up allowing improved performance and throughput.
- Allow granular security ensuring tighter control at tier/service and component level.
- Easy maintainability allowing updates to be made in isolation across services/components without affecting the entire solution as a whole.
- Increased flexibility by allowing exploitation of modular architecture of enabling systems using easily scalable components.

- Ability to separate functionally into segments where each segment can be distributed and deployed to a physically separate.

Taking into consideration the above-mentioned points, a four-tiered architecture is determined as most appropriate for Open Book.

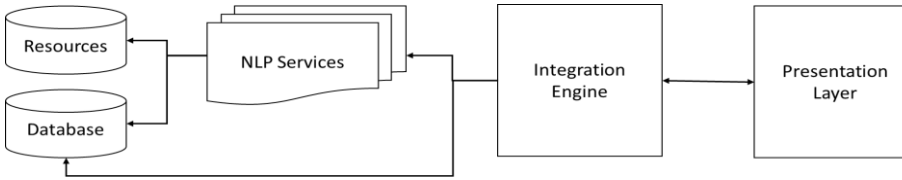

Figure 2. Four-tiered Architecture

3. Advantages

The following are the significant advantages the 3-Tier architecture gives to the Open Book solution:

- Loose Coupling: Open book architecture helps to make maintenance and enhancements easier due to the low coupling between layers, high cohesion between the layers, and the ability to switch out varying implementations of the layer interfaces.
- Maintenance: The architecture helps to control and encapsulate the complexity of large application by dividing the related functionality into different layers. It improves the maintainability of application and a well-defined set of criteria to group the functionality of the solution into a set of layers and define the services that each layer provides.
- Scalability: Distributing the layers over multiple physical tiers can improve scalability, fault tolerance, and performance.
- Performance: Architecture makes it easier to scale out when necessary to improve performance. Layered approach also helps to add multiple nodes to offer load balancing.
- Reusability: Other solutions can also reuse functionality exposed by the various layers, especially if the layer interfaces are designed.
- Testability: Testability benefits from having well-defined layer interfaces as well as the ability to switch out various implementations of the layer interfaces.

4. Tiers

4.1. Database and NLP Resources

We store user accounts, user preferences and personalized documents. Data is organized into a relational database. We use Microsoft SQL Server, version 2008 or later. The Integration engine uses Entity Framework and ADO.NET to access the database.

Open Book utilizes various resources for its operations – dictionaries, thesaurus, image galleries. Dictionaries and thesaurus are used to obtain simplified definitions of complex words and metaphors. The dictionary resources are extensible – third-party dictionary services can be hooked-up in the system to extend it with richer resources. Images are used to provide visual cue about complex words and concepts. Currently, images are retrieved using third-party services like Google and Bing.

4.2. Integration Engine

The integration engine is a middle-tier component which acts as the interface between the Presentation tier, and resources including database and partner services. Some of the features of the integration engine are:

- Act as a single point of interaction for the presentation tier
- Encapsulate complex orchestration logic exposed via a simple interface. This allows the presentation tier to act as a "thin" client.
- Integrates all NLP services into a well-defined workflow, and takes careful considerations to validate the input and output during data-exchange between these services.
- Remove tight coupling of sequential logic to be situated in the integration engine instead of the presentation tier.
- Make use of "rich" features such as caching, logging and error management ultimately improving overall system performance and user experience.
- Ability to easily extend/maintain solution reducing "ripple" change effect. Changes to business logic can be made easily, quickly and independently within the integration engine without any impact and further changes to the presentation tier.

Cross Cutting Concern

The integration engine also contains common functionality that spans multiple layers. This functionality typically supports operations such authentication, authorization, caching, communication, exception management, logging and

instrumentation, and validation. In order to ensure such functionality is not duplicated across multiple layers; this will be centralized in one location.

The Integration engine is developed as ASP.NET application using Microsoft .NET Framework. It exposes SOAP-based API for the presentation layer or for third-party tools. It uses also SOAP to communicate with the NLP services. SOAP is selected for its reliability and support for complex data types and exception handling.

4.3. NLP Services

All NLP services are implemented in Java. They exchange data between themselves using GATE XML format [4]. They expose their application programming interface via SOAP-based web-services.

GATE Document Generator

Input documents from users which may come in various formats: plain-text, PDF, Microsoft Word, and HTML. This service converts users' documents into GATE XML format documents, which are then processed by the other NLP services.

Syntax Processor Web Service

This service processes the input document to detect syntactic complexity. It tries to determine the underlying structure of complex sentences, the type and extent of conjoins and subordinated constituents. The service rewrites complex sentences into shorter, simpler sentences. The input and output of the service are both a document in Gate XML format. The rewritten sentences are exported as AltSentences.

Figurative Language Web Service

This service tries to identify figurative language expressions which are difficult to understand by people with ASD. On successful detection, the service replaces these expressions with appropriate dictionary definitions or synonymous terms. Again, the service takes a document in GATE XML format and outputs a document in the same format with injected definition for the detected figurative expressions and an additional attribute called confidence level, which reflects the trust the service has in the successful detection of the figurative expression.

Anaphora Resolution Web Service

As its name suggests, this service tries to resolve and remove anaphora from the input text. The service works in three steps:

A. Anaphora detection.

B. Potential Candidate detection: for each detected anaphor, a list of candidate antecedents is created.
C. Anaphora removal: the service tries to replace each detected anaphor with its correct antecedent.

The service takes a document in GATE XML format and outputs a document in the same format with replaced anaphora.

Image Labelling Web Service

The function of this service is to detect difficult concepts and provide corresponding images which might help better illustrate these concepts. Images are retrieved from an image database or external image providers such as Google and Bing. The service takes a document in GATE XML format and outputs a document in the same format with a special annotation set with new features added to it. These new features include the image resource location and level of confidence for the suitability of the image as illustration.

Personalized Document Generator

This service is the last process of the NLP chain. It converts the GATE XML format document into special markup format, based on HTML5. The output document contains markup annotations for every word and sentence in the output document. It relies heavily on custom data attributes [5] to embed additional information such as original sentences, and detected reading obstacles.

By using HTML5, it is guaranteed that the output document can be rendered using standard instruments, and no special rendering engine will be required. Output can be formatted for visual presentation using appropriate CSS, and exported to other formats such as PDF or Microsoft Word.

4.4. Presentation Layer

The presentation layer is the user interface component that the end users of the Open Book solution will interact with. It focuses on personalization and the specific requirements for people with ASD as described by Pavlov [6]. The presentation layer is developed as a web-based thin-client using HTML5 and JavaScript. It also has a small backend component based on ASP.NET MVC framework, and using SOAP-based messaging to communicate with the integration engine. This backend facilitates to communication with the SOAP-based interface of the Integration Engine and exposes RESTful services to the JavaScript front-end.

5. Conclusion

Open Book is a distributed software system employing various natural language processing components to simplify documents for people with ASD. A four-tier architecture is used to achieve adequate separation of concerns, and establish an extensible and reliable infrastructure between the components of the tool. The components themselves are developed in different technologies – Java and .NET Framework. This requires using standard communication and serialization formats to ensure interoperability, such as SOAP.

Acknowledgements

This work is partially funded by project IT 15-FMIIT-004 project of the Scientific Fund of the University of Plovdiv "Paisii Hilendarski", Bulgaria, and by the European Commission under the Seventh Framework Programme (FP7-2007-2013) for Research and Technological Development under grant agreement № 287607.

I would like to thank all the partners from the consortium working on the FIRST project for their valuable contribution to this work.

References

[1] Open Book Tool, http://www.openbooktool.net

[2] FIRST Project Official Web-Site, http://first-asd.eu/

[3] The Autism Book: What Every Parent Needs to Know About Early Detection, Treatment, Recovery and Prevention, Robert Sears, 2009.

[4] Cunningham, et al. Developing Language Processing Components with GATE Version 8. University of Sheffield Department of Computer Science. 17 November 2014.

[5] W3C, HTML5 Specifications, HTML5 Elements, http://www.w3.org/TR/2010/WD-html5-20101019/elements.html#attr-data. Visited October 8, 2015.

[6] Pavlov, N., User Interface for People with Autism Spectrum Disorders, *Journal of Software Engineering and Applications*, Vol. 7, No. 2, 2014, Article ID: 43152, 7 pages.

Faculty of Mathematics and Informatics
Plovdiv University “Paisii Hilendarski”
236 Bulgaria Blvd,
Plovdiv 4003, Bulgaria
E-mail: nikolayp@uni-plovdiv.bg

АРХИТЕКТУРА НА ИНСТРУМЕНТ ЗА ПОДПОМАГАНЕ НА ЧЕТЕНЕТО – OPEN BOOK TOOL

Nikolay Pavlov

Резюме. Статията описва архитектурата на инструмент за подпомагане на четенето – Open Book Tool. Представени са главните съображения при проектирането на архитектурата. Препоръчва се четирислойна архитектура и са описани предимствата на избраното решение. Статията представя компонентите на инструмента и как те взаимодействат помежду си.