COMPARATIVE EVALUATION OF GOOGLE HEALTH API VS. MICROSOFT HEALTHVAULT API

Ali Sunyaev, Alexander Kaletsch and Helmut Krcmar
Chair for Information Systems, Technische Universität München, Boltzmannstraße 3, 85748 Garching, Germany

Keywords: API, Comparative evaluation, Google Health, Microsoft HealthVault.

Abstract: Electronic Health Records (EHR) offer patients the opportunity to access their own medical records. Google and Microsoft recently extended their public services by introducing internet-based personal healthcare information platforms – Google Health and Microsoft HealthVault. Over one hundred thousand people have registered at the two services since they were launched. Both companies invite other for-profit companies as well as non-profit organizations to participate in the design, development, and distribution of their own healthcare-related applications. Such applications are based on the free accessible EHR systems of Google and Microsoft and provide further benefits to patients. Due to its simplicity and usability, an API design could determine the variety of value-added applications developed and thus be essential for the commercial success (and potential market dominance) of one of these EHR systems. This work examines and compares the designs of Google Health API and Microsoft HealthVault API. Such an evaluation provides benefits for both research and practice: on the one hand, the results provide an overview of the different open API designs, and, on the other hand, the results provide the developer community with useful lessons learned from comparing the examined APIs.

1 INTRODUCTION

Health care systems are in constant transformation. Currently, Germany is introducing an Electronic Health Card, following the global trend towards full digitalization of patient medical information. Patients' files once handwritten by medical staff will now be collected as Electronic Health Records (EHR). This transition provides numerous benefits for patients, health insurance companies and medical staff (Tang et al., 2006). For example, patients gain detailed access to their records, which they can share with family members or the doctors of their choice, and insurance companies save money by avoiding repetition in patient care.

According to a study of Roland Berger, the secondary healthcare market in Germany alone has a value of EUR 60 billion (Berger, 2007). Because of its value and growth potential, numerous solution providers are preparing to enter the healthcare market worldwide. Frameworks for EHR are an especially exciting opportunity for obtaining a strong market position. They enable third parties to develop applications in order to reach a large number of patients; e.g., a potential application could look at a patient's medications and, according to the patient's postal address, construct a database of low-cost pharmacies in order to buy the cheapest medication available and ship it to the patient. Implementing such an application and selling it to the users of the EHR systems is one possible business model for such a value-added application.The patient can choose his favoured out of a wide array of available services and add them to his personal EHR profile. Therefore, EHRs are very powerful and valuable tools for future health service providers (Sunyaev et al., 2010).

But EHR usage does not sell itself; confidence in the EHR must be developed with application providers and patients. Also, handling and usage of EHRs must be learned by both sides, as it is not always intuitive. Especially on the application providers' side, the developers play an important role. They have to handle the Application Programming Interface (API) provided by the EHR. The API therefore has to be very elaborately designed in order to gain their acceptance. This means that it has to be easy to use, well documented with good samples, and provide libraries for common programming languages. There should be

Sunyaev A., Kaletsc A. and Krcmar H. (2010).
COMPARATIVE EVALUATION OF GOOGLE HEALTH API VS. MICROSOFT HEALTHVAULT API.
In *Proceedings of the Third International Conference on Health Informatics*, pages 195-201
DOI: 10.5220/0002694901950201

no gaps or pitfalls which could pose obstacles to developers during their first contact with the API. Of course, the variety and multiplicity of functions and options available is very important, but, in the end, the simplicity and usability of an API decide whether it will be well accepted in the market, and ultimately whether or not the EHR will be a success or failure.

2 GOOGLE HEALTH

"Google Health allows you to store and manage all of your health information in one central place. And it's completely free. All you need to get started is a Google username and password" (Google, 2009a). Google Health is an EHR built to provide an easy way for users to store their own personal health records online. Google describes its product as an EHR "of a different model" which, in addition to offering a place to store, manage and share health information, also provides a directory of online services to aid in using this information on a daily basis (Google, 2009b). Such a platform strategy means patients will be able to automatically import their records, prescription history and test results, interact with services and tools such as appointment scheduling, prescription refills and wellness tools from third-party providers as they are added to the directory.

Google Health is based on open standards (Continuity of Care Record for data exchange, SOAP for the web-services interoperability), and provides a development API, programming libraries and test infrastructure. Although not an HIPAA covered entity, Google guarantees it will protect the privacy of the information by giving the user complete control over it. To this end, Google Health features no advertising. Google Health is oriented towards the U.S market, as the third-party services it uses are exclusively American.

Google Health enables third parties to contribute further services to the Google Health platform that can work with the user's data (Sunyaev et al., 2010). Google Health currently has more than ten partners who provide services for importing medical records, exploring medications and treatments, converting paper records, finding personalized tools, copying files, and sharing users' records.

3 MICROSOFT HEALTHVAULT

"HealthVault offers you a way to store health information from many sources in one location, so that it's always organized and available to you online" (Microsoft, 2009). With more than thirty partners, HealthVault also offers a broad variety of services to users. HealthVault consists of two distinct products – an electronic repository for health data and a specialized search engine for health information on the World Wide Web, both free to users (Cross, 2007). HealthVault is sometimes described as "PayPal for health information" (Blankenhorn, 2008, Berndtson, 2008, Kolakowski, 2009) for being able to store and share medical information at the discretion of its owner, as well as for utilizing similar security features. HealthVault stands out from other EHR providers because of its extensive partner network, particularly in the area of medical and fitness devices (Sunyaev et al., 2010). Microsoft plans to make money by placing ads next to the HealthVault search results. Similar to Google Health, Microsoft offers an open API and an SDK, including libraries for Java and Ruby. Microsoft HealthVault is currently U.S-centric, as it can only be used from inside the U.S and cooperates only with American hospitals, physicians and pharmacies.

4 COMPARATIVE EVALUATION

In March and April 2009 this analysis was conducted from a developer's point of view. While building an application able to send and receive medication information to and from the two solution providers (Google Health and Microsoft HealthVault), we encountered various problems, but also made some interesting discoveries. We found out that a well-designed API is not the only essential element needed for a developer to start working with a new service. The overall environment of the API matters.

Therefore, all found issues were clustered into seven different categories, which were then discussed and rated in an expert group. As fully implemented *libraries* simplify the developer's tasks considerably, their availability and quality are the first elements that must be examined in the comparative evaluation. Of course, they must be well documented and enriched with elaborate samples. Like the libraries, the API itself requires a thought-out *documentation*. As there are various kinds of applications (such as desktop or web applications), there are different requirements

regarding *authentication* for an EHR platform. Not just the authentication, but also the entire communications structure between applications and service providers must be secure. Therefore, *security* is also an important element to consider. Related to the security of connection is the safety of the user. It is very important that online stored medical items like medications, allergies or weight cannot be misused. Therefore the user should have the possibility to give very fine-grained permissions; the type of *Data Access* is the next issue in the comparison. *Data Modification* can be done in different ways, making it necessary to compare these methods. Last but not least, the assembling and usage of *Data Messages* and resulting objects are considered.

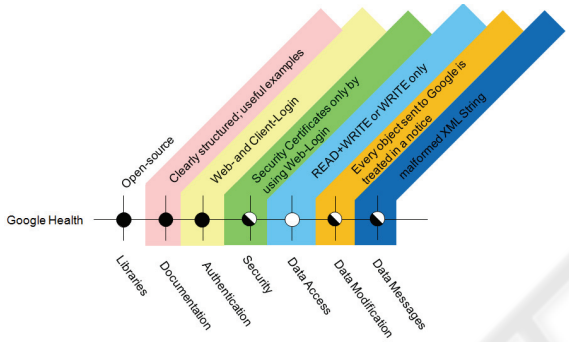

Figure 1: Google Health rating.

Figure 1 and Figure 2 show an overview of the results of the comparison. Filled circles represent excellent realisation of the corresponding issue, semi-filled represent acceptable implementation, and blank circles highlight problems.

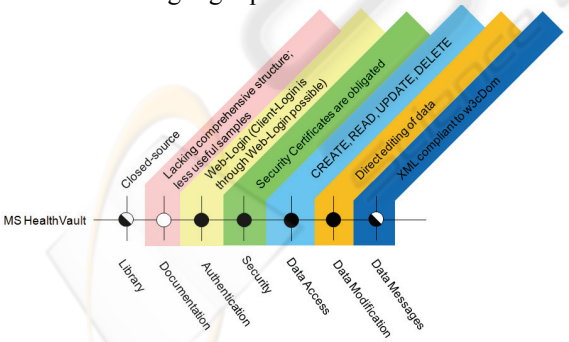

Figure 2: Microsoft HealthVault rating.

4.1 Libraries

Google Health offers a high diversity of libraries for its API in different languages. Fully implemented libraries in Java, .NET, PHP and Python exist. They are enriched with guides and adequate samples. Each library brings its own documentation with well-appointed and eligible samples with a ready-to-run code. Regarding the fact that the Google Health Data API is based on Google's Data API, (already technically matured), Google's experienced developers will gain a high recognition effect when starting new with Google Heath API. Therefore, it is quite simple to get in touch with Google Health and to build up first applications in a short period of time with use the PHP and Java library.

In contrast, Microsoft HealthVault officially offers only a .NET Developer SDK with some scattered samples. Although there are already more or less independent groups building a Java and a Ruby on Rails library for Microsoft HealthVault, both are still in an early stage of development and, of course, they are not official. Especially the scattered samples and sometimes ill-conceived guidelines make it very hard to start with Microsoft HealthVault.

As at that time we could not find any source code for the .NET libraries, we decided to try out the Java implementation. It comes along with ready-to-run examples, but building an application out of them soon becomes quite tricky because a lot of possible cases are not considered in the samples as yet, and the documentation also lacks detailed information. Further, the certificates generated by Microsoft's HealthVault Application Manager, which need to be uploaded to the platform and used with the library, are not Java compatible, and first need to be converted in a laborious process. Therefore, getting started is a lot more complicated with Microsoft HealthVault than with Google Health.

One could argue that all libraries for both systems are built on XML based frameworks, and the developer could therefore build a library on his own in whichever language is preferred. But this would be an obstacle to any plans of implementing a connection to an EHR in a short time. As a result, implementation costs would also rise without an adequate library support.

4.2 Documentation

Google Health offers a mostly well designed documentation with only a few weaknesses. The "Getting started" guide and the "Developer's" guide are helpful introductions to the API. The "Reference Guide" and library documentations offer detailed information to the developer. These clearly structured and easy to access guides, in combination with the active developer forum, establish a stable foundation for application development. This is

complemented by suitable samples included in the documentation. Ready-to-run examples exist from which a first application for testing can be developed.

Microsoft HealthVault also provides a wide variety of guidelines, but these are missing an overall structure. The website menu structure is clearly unsophisticated, requiring a lot of backward and forward browsing. Additionally, while getting started, the developers are presented information that is not useful in early stages of development. A clearer guide for developing a first application would be preferable. The time spent producing short guidance videos could be better spent creating more adequate samples. More ready-to-run code would help the user develop applications out of it.

Microsoft and Google both offer developer forums that are actively moderated by staff members. This enables developers to share problems and get feedback from an active community. In addition, online surveys are used to optimize the API.

4.3 Authentication and Security

Google Health and Microsoft HealthVault distinguish between authentications for web applications and desktop applications. While Microsoft HealthVault uses just one procedure, Google Health uses two very different procedures. In fact, the course of action for web applications from the user's perspective is almost identical for both solution providers.

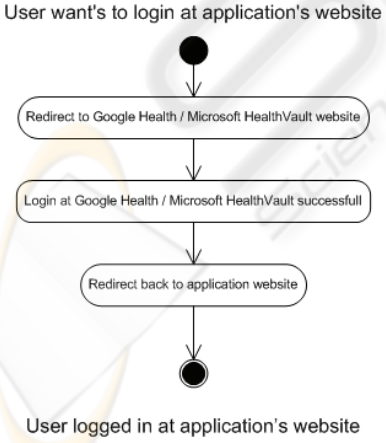

Figure 3: Web-Authentication.

Figure 3 shows systematically how this works. First, the user wants to log into the application website, e.g., the low-cost-medicine database mentioned above. Upon trying this, the user is redirected to the service provider's website, where the user can log in with his Google Health / Microsoft HealthVault account. If the user logs in successfully, he gets redirected to the application website. With this redirection a session key is transmitted, which later enables a direct authenticated connection between the application and solution provider's sites. This methodology ensures that sensitive user data, such as username and password, do not need to be transferred to the developer's site.

Google's web based method allows the developer to choose between Google's AuthSub and the open protocol OAuth in order to create a connection between a developer´s site and Google Health. Both sides are protected by domain based security certificates. As explained above, the end-user can simply log in to Google and then a session key will be exchanged between Google and the foreign site. Unfortunately, only Google Accounts can be used to log in and no other Single-Sign-On Systems are currently integrated.

For desktop applications Google offers a method called ClientLogin. This method allows the user to enter his Google login information directly into the desktop application with the information being verified online. Entering the Google username and password into a third application could be a security risk if the application is not trustworthy. Although the connection is secured by SSL and Google Health uses a valid server certificate, the client is not protected with a certificate. Therefore, the identity of the software client cannot be confirmed.

Microsoft HealthVault uses one method for web and desktop applications. Both types need to be aligned with security certificates that can be easily created and uploaded to Microsoft HealthVault by the HealthVault Application Manager. Although desktop applications are supported, they always need to be linked to an Internet domain. Hence, for a successful login a domain has to be registered at Microsoft HealthVault. On the one hand, this makes development more complicated; on the other hand, it is more secure than Google's approach. Therefore, the end-user's username and password never have to be entered at the developer's application or website. The Microsoft's HealthVault website is the one and only entry point. Also, Microsoft supports not only its own services, such as Windows Live, MSN, and Microsoft Passport, but also the authentication standard OpenID (http://www.openid.net).

While the web authentication used by the two providers is similar, Microsoft's desktop application authentication is more secure, albeit also more complicated than that of Google.

4.4 Permission Management

Microsoft HealthVault allows a detailed specification of permissions for every application installed to a profile. Create, Read, Update and Delete (CRUD) permissions can be set for every single health item class. The developer has to give reasons for every class he wants to access. Microsoft staff check the resulting document before the application can leave the test environment. While adding an application to one's HealthVault Account, the permission requirements for the application has to be accepted and is then stipulated. Therefore, once the user has accepted these, they cannot be changed afterwards. If the developer changes permission guidelines, this will not affect the user's profile at all until the user adds the application again. The detailed history function, where every action can be inspected later, is the final segment of HealthVault's excellent permission management.

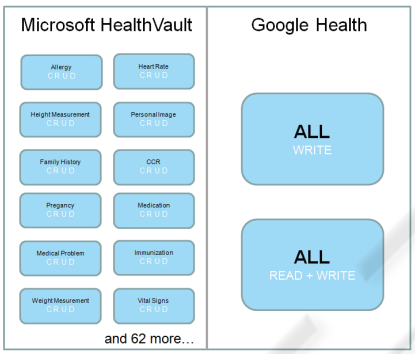

Figure 4: Granularity of Permission Management.

In contrast, Google Health allows a write-only and a read & write mode for the profile. It is not possible to set specific rights for item classes. Therefore, the user has no control over his data accessed by an application; e.g., whether an application that should only advise the user about the most affordable medications could also request personal details, such as age, sex or allergies and test results. This circumstantiality is obviously not needed to find a price for a predefined medicine, but there is actually no means of constraint. This is a serious weakness that Google should address soon. In addition, Google's history function where update notices are shown becomes quite confusing when dealing with a large numbers of events.

4.5 Data Modification

Google Health uses a notice based message system. Every medical item is packed into a notice, which is sent to Google and then added to the affected account. Only with the ClientLogin procedure is it possible to edit or delete health items. WebAuth only allows inserting. Every notice sent to Google Health is shown on the user's homepage. This ensures that users are informed of every action related to their profile.

Microsoft HealthVault's CRUD system works directly on the health items. Updating and deletion of items is always possible. This makes it easier for the developer to handle objects. But unfortunately the history function that displays changes to the user has a confusing design and is not clearly presented on the home page. Therefore, the user has to take care that he is always well informed about actions affecting his account. Considering the highly critical information that the system handles, the system should clearly inform users on the welcome page about every action related to their account in a proactive way.

4.6 Data Messages

Medical information is sent via data messages containing all the necessary information using XML format conventions.

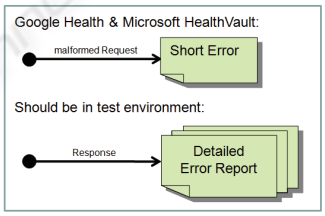

Figure 5: Desirable: More detailed error reporting.

Google Health sticks to the CCR standard. But sending standardised messages to Google Health is a little complicated initially. Unfortunately, Google's servers return only a short error notice instead of a full error report. The parser recommended by the developer's forum for XML-testing is so strict that some of Google's own samples fail. The Google health developer environment server should return full error messages, thus enabling the developer to improve the XML strings.

Receiving data messages from Google could also be improved. The libraries tested returned a malformed XML string, which has to be first repaired before it can be used in XML-parsers. Because Google's Java library's recommended function entry.getContinuityOfCareRecord().getXmlBlob().getBlob() returns a string which is missing a

surrounding XML element, a root element has to be built around the string in order to get automated parsers working. Also, the library does not offer a health object implementation that would allow the developer to avoid having to go down the entire road to the XML source.

Microsoft HealthVault unfortunately offers a lot of proprietary item sets, but also supports the CCR standard as a subset. The parallel usage of different item versions makes handling even worse. Sending data to Microsoft HealthVault is initially as complicated as sending data to Google Health. Although Microsoft fails to provide detailed error reporting, with a recommended parser from the developer forum it is possible to create valid XML-feeds in a short time. Known erroneous item sets make development more complicated and need to be worked around. The Java Framework (not officially supported by Microsoft) allows receiving data messages that are automatically converted into java objects. On the one hand, this makes developing faster; on the other hand, it is not possible to retain the plain XML easy code, which is critical.

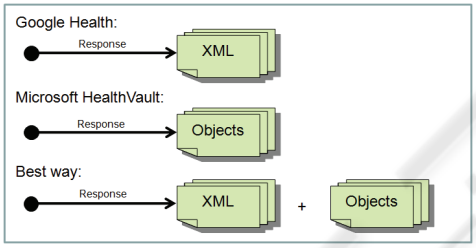

Figure 6: More alternatives in handling received data messages.

Sending and receiving data to both Microsoft HealthVault and Google Health could be improved. At the very least, the development environments should return full detailed error messages to developers. Also, the developer should be able to decide if he wants to work with objects or pure XML. Therefore, both options should be offered to developers.

5 LESSONS LEARNED

In this sub-section we discuss the important lessons learned from the development experiment by exploring several questions.

How difficult is it to get in touch with the services?
It is simple with both Google Health and Microsoft HealthVault. Just register at the developer websites, read the documentation, download libraries and get started. As we did not like Microsoft's .NET library, we had to search the web for a Java library first. Although running the supplied samples works quite well, building real applications out of them can be challenging.

How much time did it take to create the first test application?
It depended on the libraries tested. We were able to build up the first executable test cases from examples provided with Google Health's PHP and Java framework within three to four hours per language. Microsoft HealthVault's unofficial Java library took several more hours as some objects that were not as easy to adapt for our needs. Obviously, these were only test applications without any overheads like GUIs or error handling.

Did you come across any unexpected obstacles while developing your applications?
Yes, we did.

Figure 7: Web browser integrated into test application.

First example: Microsoft HealthVault encourages the developer to register a web domain that handles the login procedure. To shorten this, we integrated a web browser into our application that can fetch the session key provided by Microsoft HealthVault after the successful login. Of course, this is a simple implementation but one permitted in order to minimize testing efforts.

Figure 8: Notices in Google Health. Top: failed request (malformed CCR). Bottom: successful request.

Second example: As Figure 8 shows, the user always gets the notice, "The medical records have been copied to your profile," regardless of whether medications have in fact been copied or not. It would be less confusing for the end-user to receive the message, "No medical records have been copied."

6 CONCLUSIONS

One of the main differences between the analyzed frameworks is related to the parsing of sent or received data. "Be wary of over-specification" (Bloch, 2006), but also avoid under-specification. It should be possible to access and send data easily on a low level (XML as a string as offered by Google Health), as well as in defined cases on a higher level (e.g., medication data as provided by Microsoft HealthVault). Concerning privacy aspects, frameworks should allow users to make adjustments to the rights/privacy settings of each health item. Tracking, logging and fine-granular access rights need to be guaranteed. The use of security certificates should be mandatory. Furthermore, frameworks should rely on established standards (e.g., CCR/HL7 as offered by Google Health) instead of developing their own proprietary communication mechanisms (as it is the case with Microsoft HealthVault).

We hope that our comments will be taken as constructive criticism and will help to improve these frameworks. Our experience with these API designs confirms the importance of EHR platforms as the foundation from which promising value-added applications can be developed for future healthcare provision.

It has to be emphasized that frameworks and their environment have to be communicated in such an easy way that they are also understandable by non-experts. Although it is often not easy to explain high technical topics in common language, this should be profitable. Especially in the eHealth environment, trust must be built with all stakeholders. Trust cannot be developed while hiding behind complex terms. It has to be built on open communication and common sympathy. Therefore, persuading developers and decision makers with easy to enter frameworks is the first step in order to establish a widely used Electronic Health Record.

REFERENCES

Berger, R. 2007. Study on the secondary healthcare market. Available: http://www.rolandberger.com/company/press/releases/517-press_archive2007_sc_content/Study_on_the_secondary_healthcare_market.html [Accessed 2009-07-10].

Berndtson, C. 2008. *Microsoft, Google Joust—and Concur—On Personal Health Records* [Online]. Available: http://www.crn.com/healthcare/212101164;jsessionid=MH30505NPU55OQSNDLOSKHSCJUNN2JVN [Accessed 18.05.2009].

Blankenhorn, D. 2008. *Microsoft HealthVault is nothing like Google Health* [Online]. Available: http://healthcare.zdnet.com/?p=742&tag=rbxccnbzd1 [Accessed 03.05.2009].

Bloch, J. 2006. How to design a good API and why it matters. *Companion to the 21st ACM SIGPLAN symposium on Object-oriented programming systems, languages, and applications.* Portland, Oregon, USA: ACM.

Cross, M. 2007. Goliath moves into healthcare records. *BMJ,* 335.

Google. 2009a. *Google Health* [Online]. Available: https://www.google.com/health/ [Accessed 2009-07-08].

Google. 2009b. *Google Health FAQ* [Online]. Available: http://www.google.com/intl/en-US/health/faq.html [Accessed 24.05.2009].

Kolakowski, N. 2009. *Microsoft's HealthVault a Challenge to Google Health?* [Online]. Available: http://www.eweek.com/c/a/Health-Care-IT/Microsofts-HealthVault-A-Challenge-to-Google-Health-711489/ [Accessed 03.05.2009].

Microsoft. 2009. *Microsoft HealthVault* [Online]. Available: http://www.healthvault.com/personal/websites-overview.html [Accessed 2009-07-08].

Sunyaev, A., Chornyi, D., Mauro, C., Krcmar, H. 2010. Evaluation Framework for Personal Health Records: Microsoft HealthVault vs. Google Health. Proceedings of the Hawaii International Conference on System Sciences (HICSS 43), January 5 – 8, 2010, Kauai, Hawaii. To appear.

Tang, P. C., Ash, J. S., Bates, D. W., Overhage, J. M. & Sands, D. Z. 2006. Personal Health Records: Definitions, Benefits, and Strategies for Overcoming Barriers to Adoption. *Journal of the American Medical Informatics Association,* 13**,** 121-126.