Spreadsheet Synchronization for Workgroup Collaboration and Software Platform Access

Massimo Maresca[1,2], Pierpaolo Baglietto[1]

[1]CIPI - University of Genoa, Italy
via Opera Pia 13, 16145 Genova, Italy
[2]International Computer Science Institute
1947 Center Street, Berkeley California, USA
m.maresca@cipi.unige.it
p.baglietto@cipi.unge.it

Abstract. In this paper, we propose a new paradigm for spreadsheet-based Workgroup Collaboration and Software Platform Access. The proposed paradigm is based on the concept of Spreadsheet Synchronization, both among multiple spreadsheets and among spreadsheets and Software Platforms. Spreadsheet Synchronization in turn is at the basis of Spreadsheet Overlays, which are composite spreadsheet structures supporting the creation and the management of dynamic collaboration spaces based on the exchange of data contained within tables inside spreadsheets.We present the paradigm and compare it with the existing paradigms based on Cloud based spreadsheet sharing and on SQL enabled spreadsheets. We show that synchronization is superior to the existing paradigms based on file sharing and or on SQL queries to access data bases. More specifically synchronization provides a better level of security, controlled access and real time data support. We present a set of usage patterns and some simple use cases. We finally present a summary of the preliminary feedbacks that we obtained from a set of ongoing trials that we are currently deploying in a number of companies and public institutions.
Keywords: Workgroup Collaboration, Data Synchronization, End User Computing.

1 Introduction

Table is probably the data structure that best matches people preferences. Not only a quick look at a table allows catching the essence of a phenomenon but, in addition, a table can be easily revised, compared, integrated and combined with other tables [Grossman 2002]. The relevance of table in end user data processing is demonstrated by the diffusion of spreadsheet platforms, which are widely used in personal data analysis and presentation.

Spreadsheets belongs to a discipline called End User Programming [Nardi 1993], in which people focus on problems rather than on software technology. The huge number of End User Programmers [Scaffidi e al. 2005] and the impressive diffusion of the Microsoft Office Suite [Microsoft 2014, Covert 2013] motivate research on spreadsheet engineering [Ko et al. 2011, Erwig 2009]. Recently, spreadsheets have evolved from personal office tools aimed at improving people productivity to enterprise level tools aimed at supporting Workgroup Collaboration and access to Software Platforms. In Workgroup Collaboration, users enter data deriving from local activities, make such data of part of it available to other users, and combine the data obtained from other users with local data. As users interact with data mainly through spreadsheets, spreadsheet based Workgroup Collaboration has become a dominant paradigm, as demonstrated by the common practice of exchanging Excel attachments over email. However, such a practice has not been engineered so far. In particular, users control table exchange/update manually through copy/paste/attach-to-email/extract-from-email operations. Unsupervised data sharing and circulation often leads to errors or, at the very least, to inconsistencies, data losses, and proliferation of multiple copies (the so called “Multiple Versions of the Truth”). To improve spreadsheet-based Workgroup Collaboration, the major IT companies have introduced the spreadsheet sharing paradigm [Google 2015, Microsoft Excel 2015], which leverages Cloud computing and in particular storage delocalization and SaaS (Software as a Service) technology.

Spreadsheets also play the role of clients of Software Platforms to allow users to apply local post-processing and to create personalized presentations. Software Platforms traditionally include those in the enterprise domains and

those in the Open Data domain. In addition, the recent diffusion of Big Data analytics has introduced a paradigm in which the core processing functionalities, typically based on Hadoop, have moved to the Cloud, while many other functionalities, such as data collection, cleaning, pre-processing, and post-processing, still take place in the user laptops and PCs [Fisher et al. 2012]. The use of spreadsheets as clients of Software Platforms mainly takes advantage of the ubiquitous "Save in Excel" command, that decouples the Software Platform environment from the spreadsheet environment. Unfortunately, like spreadsheet exchange, also spreadsheet-based Software Platform Access has not been engineered so far. In particular, users extract snapshots of Software Platform data at different times and often end up working on different data versions (again Multiple Versions of the Truth). To improve spreadsheet-based Software Platform access, the major IT companies have recently empowered spreadsheets with advanced SQL query functionalities that support the periodical retrieval of data updates [MicrosoftPQ 2015, Oracle 2010].
In this paper, we propose a new paradigm to enable spreadsheet-based Workgroup Collaboration and Software Platform Access. The proposed paradigm leverages the concept of Spreadsheet Synchronization, among spreadsheets in Workgroup Collaboration, and between Spreadsheets and Software Platforms in Software Platform Access. The original idea, proposed in [Mangiante et al. 2012, Baglietto et al. 2010, Baglietto et al. 2011] in a preliminary form, has recently evolved toward Spreadsheet Synchronization. Spreadsheet Synchronization in turn is at the basis of Spreadsheet Overlays, which are composite spreadsheet structures supporting the creation and the management of dynamic collaboration spaces based on table exchange. In such spaces, people configure spreadsheet dependence relations and rely on an automatic mechanism to keep the spreadsheets synchronized. The presentation is organized as follows. In section 2 we introduce spreadsheet synchronization, in Section 3 we focus on Workgroup Collaboration, in Section 4 we focus on Software Platform Access, in section 5 we present the result of some experiments conducted in field experiments and in section 6 we provide a concluding remark.

2 Spreadsheet Synchronization

We propose a virtual space for tabular data exchange and synchronization over the Internet and over private networks (Fig. 1). Spreadsheets connect to each other over such a space, synchronize and exchange information. Synchronization is implemented as follows: a spreadsheet grants read access rights on a cell range or on a table to a set of users, and such users display an image of that cell range or table in their spreadsheets. We remember that according to spreadsheet terminology, a cell range is a fixed sixe array of cells while a table is an extensible list of records in which formulas automatically cover the new records at their insertion.
We call View a cell range or a table exposed by a spreadsheet, and Image the copy of a remote View displayed in the target spreadsheet in read-only mode (i.e the owner of the remote view keep control over data). A Cloud platform takes care of View persistence and of View-Image synchronization.
We also extend the concept of View from spreadsheets to Software Platforms by allowing any Software Platform to create data Views in the same way as spreadsheets. Thanks to such an extension, spreadsheets can maintain synchronization not only with other spreadsheets but also with external data sources.
We now introduce the concept of Service Overlay (SO), which can be defined as a set of synchronized spreadsheets and Software Platforms (Fig. 2). A SO includes both spreadsheets, which act both as sources and as targets of

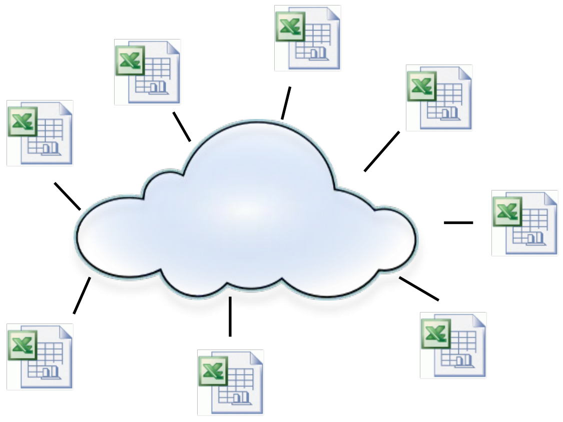


Fig. 1 - The Spreadsheet Space

information exchanges, and Software Platforms, which act as information sources and typically provide dynamic data, i.e., data evolving over time.
A SO extends the spreadsheet concept to a distributed environment and provides an innovative utilization of pervasive Internet connectivity. Spreadsheet users provide both contents, through values, and processing functionalities, through formulas. Any
cell update in a spreadsheet may trigger recalculation in other spreadsheets, result in other cell updates, which in turn may trigger other recalculations and so forth.
SO provides advanced functionalities to support Workgroup Collaboration and Software Platform Access. Section 3 and 4 describe the two cases.
Spreadsheet synchronization requires the introduction of two functionalities, respectively called *Expose* and *Join*. *Expose* is the functionality through which a spreadsheet or a Software Platform creates a View, whereas *Join* is the

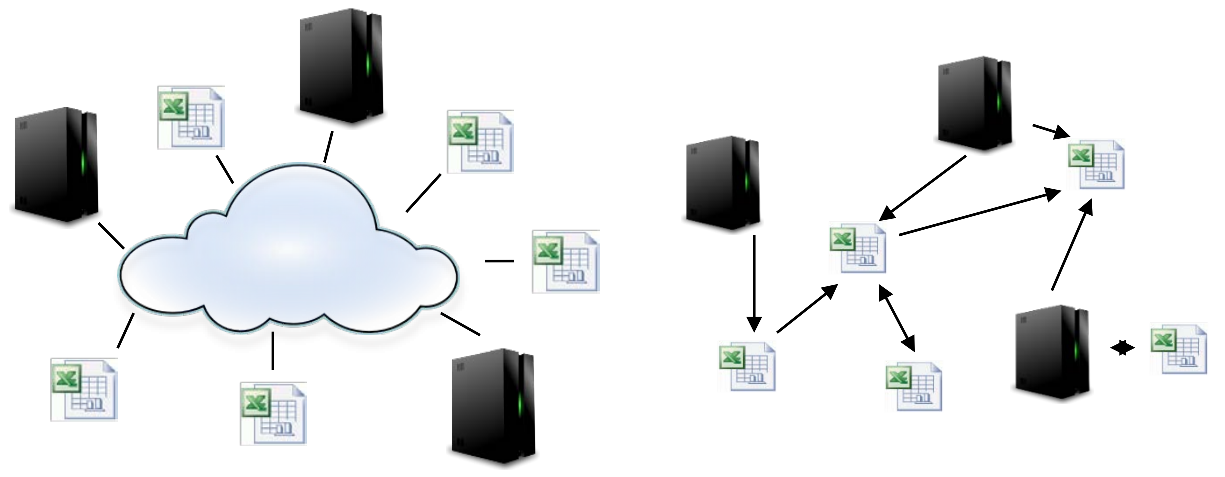


Fig. 2: Spreadsheet Overlay

functionality through which a spreadsheet creates an Image corresponding a View. *Expose/Join* create a permanent asymmetric connection from a source spreadsheet, i.e., the one that exposes the View, to the target spreadsheets, i.e., the ones that display Images of the exposed View. The “permanent” adjective denotes the fact that the View-Image relationship remains active until it is explicitly removed, whereas the “asymmetric” adjective denotes the fact that the two spreadsheets play different roles, namely the source spreadsheet owns the View whereas Images are just read-only copies of the View. Any update in an exposed View appears in the corresponding Images.
Four aspects deserve to be described in more depth.

- Formats and Styles: The formats and the styles assigned to cell ranges and tables exposed as Views are preserved in the corresponding Images. The Images appear in the same way as the Views to which they are connected.
- Formulas vs Values: Depending on configuration, either values or formulas may be are reported in Images.
- Update policy: When the value of cell range/table exposed as a View changes it is expected that the update appears in the corresponding Images. This is actually what happens, but the time at which it happens depends on how the source and the target spreadsheets have configured their operation. Both Expose, at the source, and Join, at the target, can be configured either in manual mode or in automatic mode, as indicated in the table below.

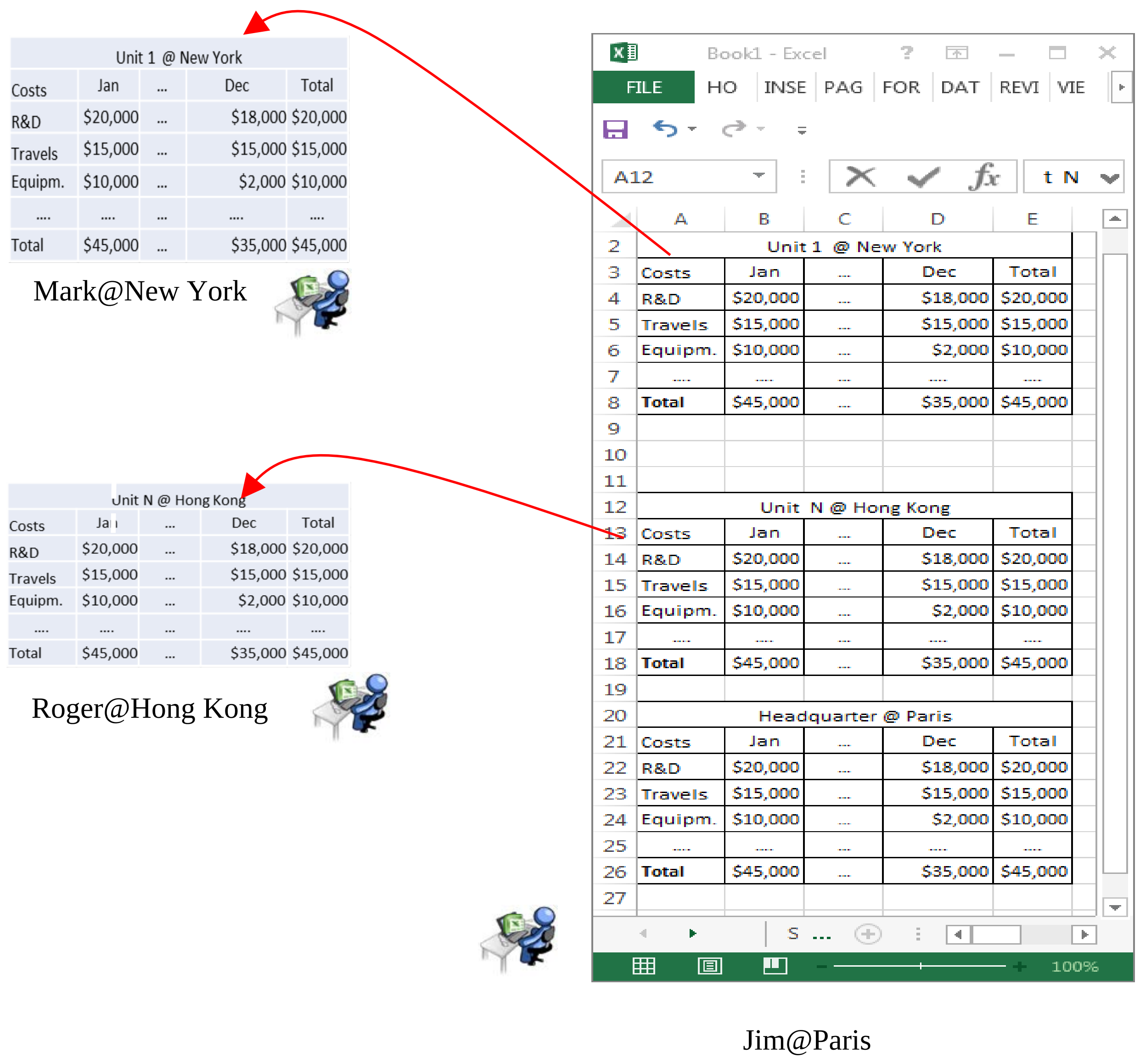
	Manual	Automatic
Expose	View update takes place upon an explicit command issued by the exposing user.	View update takes place at the update of the corresponding Spreadsheet Element.
Join	Image Refresh takes place upon an explicit command issued by the joining user.	Image refresh takes place at the update of the corresponding View.

3 Spreadsheet-based Workgroup Collaboration

Spreadsheet synchronization empowers Workgroup Collaboration by enabling the creation of customized analyses and presentations that automatically evolve with data updates in remote spreadsheets.

3.1 Usage Patterns

The following usage patterns explain how spreadsheet synchronization can be effectively used to improve Workgroup Collaboration. The first pattern is concerned with **Information Publication (**Fig. 3**)**. Information publication refers to the situation in which a spreadsheet publishes data that evolves over time, to allow other spreadsheets to incorporate such data and build dynamic analyses and presentations. When information diffusion travels over e-mail, as it usually happens, the fact that the data provided evolves over time forces the publisher to send new data versions at any update and the receiver to check the incoming mail and get the new data at any new message through multiple *copy/paste/attach/mail send/mail receive* operations.



Unit 1 @ New York				
Costs	Jan	...	Dec	Total
R&D	$20,000	...	$18,000	$20,000
Travels	$15,000	...	$15,000	$15,000
Equipm.	$10,000	...	$2,000	$10,000
....	....	...	....	....
Total	$45,000	...	$35,000	$45,000

Unit N @ Hong Kong				
Costs	Jan	...	Dec	Total
R&D	$20,000	...	$18,000	$20,000
Travels	$15,000	...	$15,000	$15,000
Equipm.	$10,000	...	$2,000	$10,000
....	....	...	....	....
Total	$45,000	...	$35,000	$45,000

	A	B	C	D	E
2	Unit 1 @ New York				
3	Costs	Jan	...	Dec	Total
4	R&D	$20,000	...	$18,000	$20,000
5	Travels	$15,000	...	$15,000	$15,000
6	Equipm.	$10,000	...	$2,000	$10,000
7	....	....	...	....	....
8	**Total**	$45,000	...	$35,000	$45,000
9					
10					
11					
12	Unit N @ Hong Kong				
13	Costs	Jan	...	Dec	Total
14	R&D	$20,000	...	$18,000	$20,000
15	Travels	$15,000	...	$15,000	$15,000
16	Equipm.	$10,000	...	$2,000	$10,000
17	....	....	...	....	....
18	**Total**	$45,000	...	$35,000	$45,000
19					
20	Headquarter @ Paris				
21	Costs	Jan	...	Dec	Total
22	R&D	$20,000	...	$18,000	$20,000
23	Travels	$15,000	...	$15,000	$15,000
24	Equipm.	$10,000	...	$2,000	$10,000
25	....	....	...	....	....
26	**Total**	$45,000	...	$35,000	$45,000
27					

Fig. 3: Information Collection

Through spreadsheet synchronization, spreadsheet users Expose Views of the data they want to publish and Join Views exposed by remote users. Joining a remote view results in the creation of an image of the view in a spreadsheet. The source users selects the spreadsheet element, indicates the list of recipients, and issues an *Expose* command. The target users select the location at which they want to place the views in their spreadsheets and issue a *Join* command. The *Expose-Join* command pair allows establishing permanent connections between the exposed views in the source spreadsheets and the images in the target spreadsheets, so as to guarantee alignment.
The second utilization pattern is concerned with **Information Collection**. Information collection refers to the very common situation in which a spreadsheet user creates a form to be filled out by other users and sends the form to such users to have them fill it out, update it and finally submit it. The possibility to update the form over time requires the exchange of successive email messages through multiple *copy/paste/attach/mail send/mail receive* operations. The form creator simply creates and formats a cell range, enters the list of the target user requested to fill out the form, sends them the cell range and requests them to Expose it.

3.2 Use Cases

Product list management is an example of *Information Publication*. A product list manager maintains a product list in a central Excel spreadsheet and publishes the regional product lists, exposing each regional list to the corresponding regional area manager. The regional managers link their spreadsheets to the central spreadsheet and create personal analyses and presentations. As the regional manager spreadsheets and the central spreadsheet constantly maintain alignment, the personal analyses and presentations automatically follow the evolution of the central product list (e.g., product price variation, stock availability, etc).
Budgeting is an example of *Information Collection*. A budget coordinator collects budget data from the contributing organizational structures (e.g., procurement, human resource, research, etc.), which in turn collect data from lower level organizational structures. The global budget can be seen as a live composition of data evolving over time, typically following top-down directives from the budget coordinator to the contributors and bottom-up data updates from the contributors to the budget coordinator.

4 Spreadsheet-based Access to Software Platforms

Spreadsheet users often use spreadsheets to integrate and process data extracted from a variety of IT Platforms. As extraction takes place under user control and has as a result an instant picture of the Software Platform data, different users extracting the same data at different times may end up working on inconsistent data versions. Moreover, each data update on the user desktop requires a new extraction and a full download, independent of the fact that the update may affect only a small fraction of the data extracted, which may result in a potentially heavy impact on network and computing resources, depending on the size of the data extracted and on the update frequency.

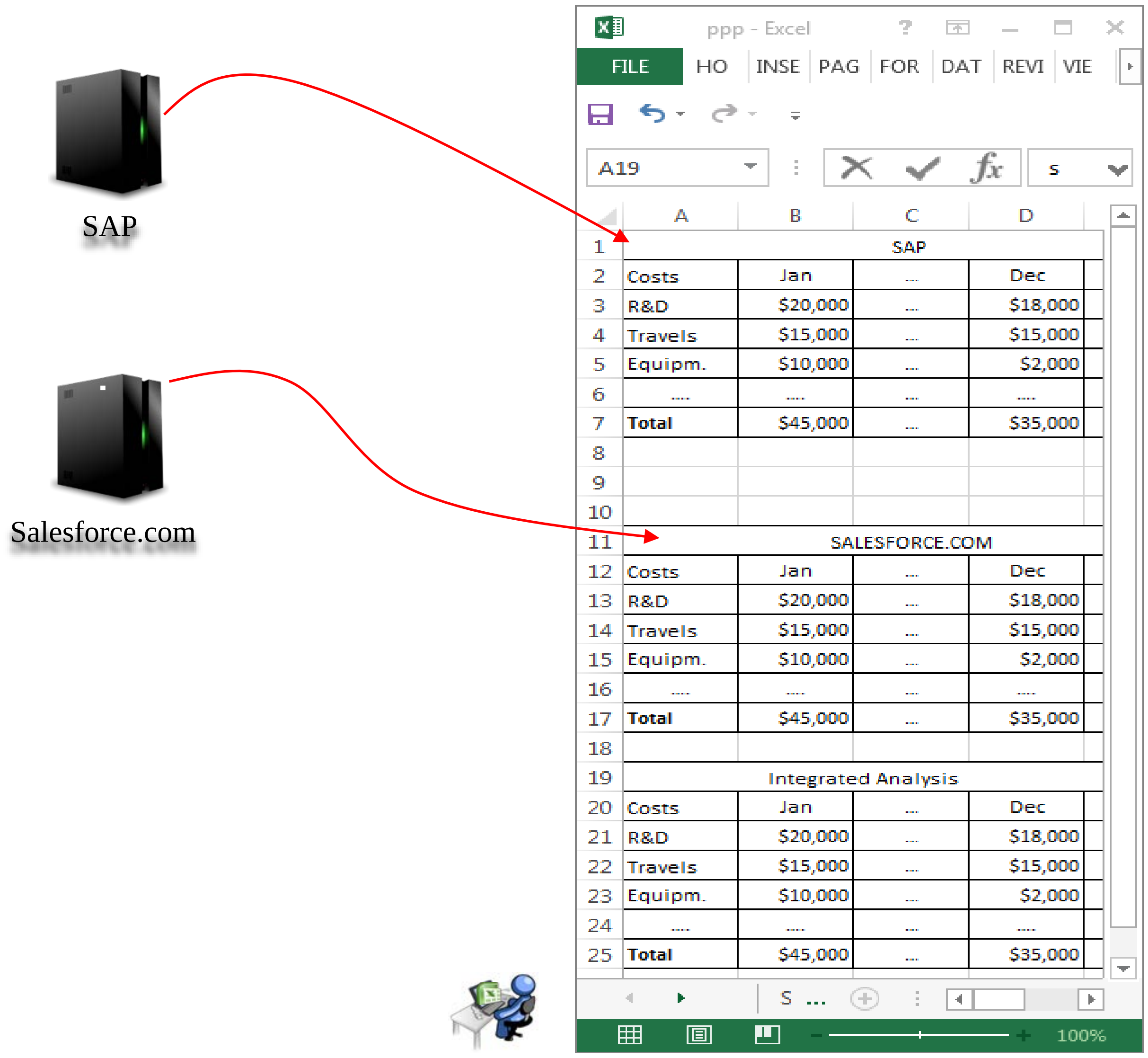
4.1 Usage Pattern

Spreadsheets natively offer a set of functionalities to connect to external information sources and databases. Spreadsheet synchronization proposes a different usage pattern. Instead of assigning spreadsheet users the task to query the Software Platforms, the Software Platform administrator creates views of corporate data whereas spreadsheet users only join such Views (Fig. 4). Leveraging synchronization with multiple Software Platforms, spreadsheet users can integrate the information stored in such platforms create customized analyses and presentations that automatically evolve with the corporate data. Spreadsheet synchronization allows companies to create “live” Views of corporate data and allows corporate Excel users to maintain local Images of such Views. Such Images can be used as inputs in formulas/functions to create personalized analyses and can be assigned styles and formats to create personalized presentations. Presentations are “live”, as data updates immediately appear in the user worksheets and trigger the automatic refresh of the personalized analyses and presentations that depend on them.

4.2 Use Cases

Personal Decision Support Systems is the most general use case of spreadsheet synchronization with Software Platforms. While company IT Platforms typically offer built-in Business Intelligence functionalities, managers and executives often prefer to develop customized analyses and graphical presentations in spreadsheets, an environment they know well and that provides them with a powerful set of tools. For example, they develop and share what-if analyses or rough statistical calculations on small subsets of the corporate data to support their strategy options.

Enterprise Application Integration is the second use case of interest. The presence of multiple autonomous IT Platforms forces Managers and Executives to integrate the corporate data exposed by the different IT Platforms on their desktops. Spreadsheets become powerful data mashup tools facilitating the integration of data from multiple origins.



	A	B	C	D
1		SAP		
2	Costs	Jan	...	Dec
3	R&D	$20,000	...	$18,000
4	Travels	$15,000	...	$15,000
5	Equipm.	$10,000	...	$2,000
6	....	....	...	....
7	**Total**	$45,000	...	$35,000
8				
9				
10				
11		SALESFORCE.COM		
12	Costs	Jan	...	Dec
13	R&D	$20,000	...	$18,000
14	Travels	$15,000	...	$15,000
15	Equipm.	$10,000	...	$2,000
16	....	....	...	....
17	**Total**	$45,000	...	$35,000
18				
19		Integrated Analysis		
20	Costs	Jan	...	Dec
21	R&D	$20,000	...	$18,000
22	Travels	$15,000	...	$15,000
23	Equipm.	$10,000	...	$2,000
24	....	....	...	....
25	**Total**	$45,000	...	$35,000

Fig. 4: Spreadsheet based Access to Software Platforms

Alignment with customers and providers is a third use case of interest. Companies may provide live data views to customers to allow them to develop personalized analyses and graphical presentations. For example, a bank/broker/stock-exchange may give its customers real time views on stock quotes, bond quotes, currency exchange rates and other financial figures, in such a way to allow them to build live portfolio analyses and graphical presentations. Alternatively, companies may export corporate data to their service providers to give them the elements they need for service implementation. For example, a company may export the employee's daily activity reports to its HR management service provider and information on invoices and bank cash flow to its accountant.

5 Field Experiments

A test system is available as a free service over the Internet see (www.spreadsheetspace.net) for Microsoft Excel users. The platform includes a Spreadsheet Client and a Cloud Server. In addition, it offers a Web Service Interface for those who want to implement a connector and a management console.

The Spreadsheet Client exploits COM Excel Addin and C# technology. It offers a standard ribbon interface through which Microsoft Excel users can activate the synchronization services. The Cloud Server exploits Open Source Java technology and runs in the Amazon EC2 Cloud. The Web Service Interface allows application developers to create software platform connectors using standard interfaces based on REST Web Services and JSON data format. Finally, the Management Console allows users to configure the synchronization service.

We ran several field experiments, in large/small companies and public administrations. The following list summarizes the indications that we obtained from such experiments and that contributed to the development and to the engineering of the platform. We classify such indication in three categories, namely indications related to the Spreadsheet Client, indications related to the Cloud Server, indications related to Operation Administration and Management, indications related to functionalities:

Indications related to the Spreadsheet Client

- Installation: The Spreadsheet Client must be installed/uninstalled using standard procedures and with no administrator privileges, to allow enterprise users to bypass the restrictions that the company IT management team often imposes over third party software installation.
- Firewall/Proxy traversal: The Spreadsheet Client must include appropriate procedures to connect to the Cloud Server from private networks protected by firewalls and proxies in a transparent way. In addition, it must include a mechanism that allows receiving the events issued by the Cloud Server in spite of the fact that the Spreadsheet Clients are typically located in private administrative domains and are unreachable from outside.

Indications related to the Cloud Server

- Public Cloud server: Scalability imposes that the public Cloud Server must play the role of a View router. All operations, in particular compression and encryption, must take place in the Spreadsheet Clients whereas no operation must take place in the Cloud Server.
- Private Cloud server: To support full data privacy the Spreadsheet Server must be deployable in enterprise data centers and private administration domains instead of using the public service available in the Cloud;

Indications related to Operation, Administration and Management

- User Roles: We have come up with a classification of the people that take advantage of spreadsheet synchronization based on the following roles:
    - User: refers to the people who expose/join Views through the spreadsheet GUI;
    - Programmer: refers to the people who expose/join Views through the spreadsheet programming interface;
    - Connector Manager refers to the people who create the Views exposed by Software Platforms;
    - Data Provider refers to the people who populate the Views exposed by Connector Managers;
    - Platform Instance Manager refers to the people who manage the Platform in organizations;
    - SO Coordinator refers to the people who design, deploy and control the Spreadsheet Overlays;
    - SO Participant refer to the people who participate in Spreadsheet Overlays.
- Management Consoles: A set of management consoles tailored to the different user roles must provide the appropriate tools for View/Image/Form management as well as for SO management, user management and performance management.
- Monitoring: The Cloud Platform must log the Spreadsheet Client presence, i.e., activation and reachability, the Spreadsheet Client activity, i.e., View Exposes, Joins, Updates, and Image Refreshes, and the Cloud Server Activity. All the events, appropriately time-stamped, must feed both traditional textual logs and live dashboards for the different user roles.

Indications related to functionalities

- Versioning: As manual synchronization allows exposers, both spreadsheets and software platforms, to create successive versions of the same data, a mechanism to track the data versions and to navigate over such versions is mandatory. “Undo Update/Refresh” is a first instance of such a mechanism, in that it allows browsing a View update to decide whether to accept it or not.
- Send/Receive: Pilot users requested a simple functionality to send/receive spreadsheet elements to/from other users without leaving the spreadsheet. Although such a functionality seems to be already covered by email, users pointed out that its direct support in spreadsheets facilitates people collaboration and eliminates the errors due to cut/paste/switch to email etc.
- Other functionalities: Two additional functionalities based on spreadsheet synchronization are currently under investigation. Such additional functionalities, not included in the current pilot, include:
    - **Selective Join**, to join only a row of a View of a table instead of joining the entire table. Such a functionality was proposed to subscribe to an item of a list, e.g., a stock record in a stock record list;
    - **Share**, to support spreadsheet element co-editing. Such a functionality does not differ from the one offered by Google Spreadsheet [Google 2015] and Zoho [Zoho 2015], but it is based on desktop processing and as a consequence supports privacy.

6 Summary and Concluding Remarks

In this paper, we have proposed spreadsheet synchronization as the main paradigm to support Workgroup Collaboration and Software Platform Access. We implemented a software platform to support this paradigm (see www.spreadsheetspace.net) and tested it in laboratory and with a number of pilots.

The Spreadsheets synchronization paradigm allows users to keep working on their fully featured desktop version of MS Excel while collaborating through the exchange of data. This paradigm also allows to overcome some problems related to the Excel-to-Software Platforms connections, in particular the non-real-time and the scalability problems of the built-in “SQL query” and similar functionalities.

Finally we claim that cloud based sharing does not support information confidentiality while synchronization based sharing does. This is a key point for sustainable data exchange especially in B2B applications (see the www.industrialdataspace.com initiative for example) where data ownership and confidentiality must be assured.

7 Bibliography

Baglietto P., Fornasa M., Mangiante S., Maresca M., Parodi A., Stecca M. 2011. A Platform for Service Composition, Proc. European Spreadsheet Risks Int. Grp. (EuSpRIG), Greenwich, UK.

Covert A. 2013. Will Google Docs kill off Microsoft Office?, CNNMoney (New York), http://money.cnn.com/2013/11/13/technology/enterprise/microsoft-office-google-docs/

Erwig S. 2009. Software Engineering for Spreadsheets, IEEE Software, Vol. 26, N. 5, pp. 25-30.

Fisher D., DeLine R., Czerwinski M, Drucker S. 2012. Interactions with Big Data Analytics, ACM Interactions, Vol. XIX (3) May-June, 50-59.

Google 2015. Google Docs, http://www.google.com/docs

Grossman T. 2002. Spreadsheet Engineering: A Research Framework, *Proc. European Spreadsheet Risks Interest Group Symposium*, Cardff, Wales, July.

Ko A.J., Anraham R., Beckwith L., Blackwell A., Burnett N, M., Erwig M., Scaffidi C., Lawrance J., Lieberman H., Myers B., Rosson M. B., Rothermel G., Shaw M., and Wiedenbeck S. 2011. The State of the Art in End User Software Engineering, ACM Computing Surveys, Vol. 43, pp. 21-44.

Mangiante S., Maresca M., Roncarolo L. 2012. SpreadComp platform: A new paradigm for distributed spreadsheet collaboration and composition, Proc. 8th International Conference on Collaborative Computing: Networking, Applications and Worksharing, Pittsburgh, PA, USA.

Baglietto P., Cosso F., Fornasa M., Mangiante S. , Maresca M. , Parodi A., Stecca M. 2010. Always-on Distributed Spreadsheet Mashups, Proceedings of the 3rd and 4th ACM International Workshop on Web APIs and Services Mashups, Article No. 8.

Microsoft by the Numbers 2014. http://news.microsoft.com/bythenumbers/index.html.

Microsoft, 2015. Introduction to Microsoft Power Query for Excel, https://support.office.com/en-in/article/Introduction-to-Microsoft-Power-Query-for-Excel-6e92e2f4-2079-4e1f-bad5-89f6269cd605

Microsoft, Microsoft Excel Online, 2015, https://office.live.com/start/Excel.aspx,

Nardi B. A. 1993. A Small Matter of Programmig, Perspectives on End User Computing, MIT Press.

Oracle, 2010. Oracle® Hyperion Smart View for Office Fusion Edition, Overview, http://www.oracle.com/technetwork/middleware/smart-view-for-office/overview/smart-view-overview-wp-134759.pdf

Scaffidi C., Shaw M., and Myers B. 2005. Estimating the number of End Users and End User Programmers, Proc. IEEE Symp. Visual Languages and Human Centric Computing (VLHCC '05), pp. 207-214, 2005.