# Integrating Modeling Languages and Web Logs for Enhanced User Behavior Analytics

Carlo Bernaschina, Marco Brambilla, Thanas Koka, Andrea Mauri, Eric Umuhoza
Politecnico di Milano, DEIB.
Piazza Leonardo da Vinci, 32
I-20133 Milan, Italy
name.surname@polimi.it

## ABSTRACT

While basic Web analytics tools are widespread and provide statistics about Web site navigation, no approaches exist for merging such statistics with information about the Web application structure, content and semantics. We demonstrate the advantages of combining Web application models with runtime navigation logs, at the purpose of deepening the understanding of users behaviour. We propose a model-driven approach that combines user interaction modeling (based on the IFML standard), full code generation of the designed application, user tracking at runtime through logging of runtime component execution and user activities, integration with page content details, generation of integrated schemaless data streams, and application of large-scale analytics and visualization tools for big data, by applying both traditional data visualization techniques and direct representation of statistics on visual models of the Web application.

## Keywords

Web analytics; model-driven development; user interactions modeling; user behavior analysis; IFML

## 1. INTRODUCTION

In recent years, the software language engineering community has put more and more emphasis on the design and experimentation of languages that cover the requirement specification [2, 1], design [13, 5], and verification/validation [9, 7, 8, 6] of software artifacts. Some of these experiences have also spawned commercial products [11, 12] and thus have been applied in industrial settings, with excellent results.

On the other side, a completely different line of work is ongoing regarding the usage analysis of Web applications with the aim of extracting leads to optimizing the user experience. Indeed, with the increasing need to meet customer preferences and to understand customer behavior, Web analytics has become the tool of choice towards taking informed business and interaction design decisions. Several tools exist that support analysis of Web server logs and extract information on application usage (based on data mining, machine learning, and big data analysis; or simpler pragmatic approaches in the UX field, like A/B split testing). However, those tools are unaware of the design structure and the actual content managed by the application. Therefore, understanding the relation between the output of the quantitative log analysis, the structure of the application, and the displayed content is not an easy task, especially for large and complex systems. Some analytics tools, like Google Analytics [10], can take into account the page content but this implies a high development overhead which yields to high risk of errors and maintainability.

In this paper we propose a model-driven engineering approach that combines user interaction models with user tracking information and details about the visualized content in the pages. Our claim is that integration of appropriately designed modeling languages for user interaction development and Web log analytics approaches has high potential of delivering valuable insights to designers and decision makers on the continuous improvement process of applications. We show our full development cycle from visual application modeling (through the OMG's standard Interaction Flow Modeling Language [4, 3]), to full code generation, user interaction log collection and integration with content and semantics of visualized data, down to display of results both through traditional data visualization techniques and with direct representation of statistics on visual models of Web applications represented with the OMG's IFML standard. Our proposal can be applied to applications developed using any model-driven development approach and generating runtime usage logs containing references to the conceptual model elements.

## 2. MODELING, INTEGRATION, ANALYSIS, AND VISUALIZATION

In this section we summarize the steps of our method, as reported in Figure 1.

### 2.1 Application Modeling and Generation

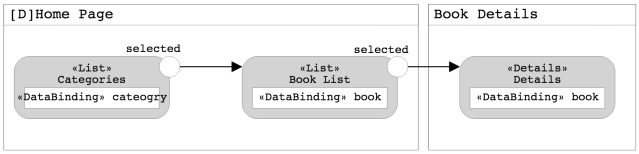



**Figure 2: Example of IFML diagram.**


*WWW'17 Companion,* April 3–7, 2017, Perth, Australia.
ACM 978-1-4503-4914-7/17/04.
http://dx.doi.org/10.1145/3041021.3054734

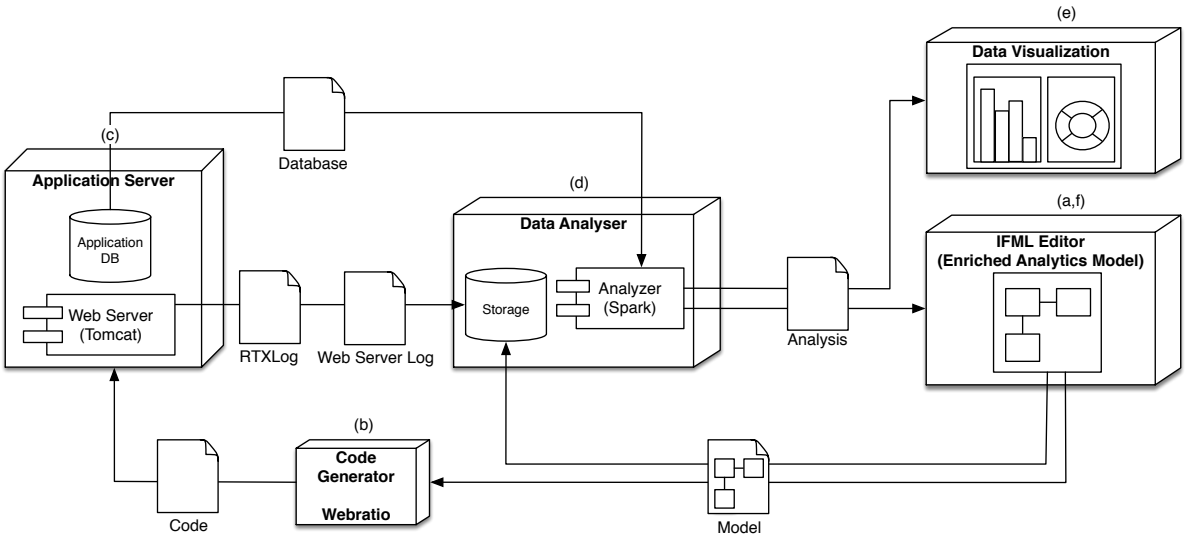



**Figure 1: Architectural overview of the approach and flow of artifacts.**

The first phase consists on modeling the web application using a visual modeling editor supporting the IFML standard (Figure 1 (a)). The designed model describes both the organization of the persistence layer in terms of UML class diagram or ER diagram (where the core information is *entities*, *attributes* and *relationships*); and the interaction flow model. The latter describes the user interaction of the application as: (i) the composition of the Web application, in terms of web pages (*ViewContainers*) and widgets (*ViewComponents*) contained into the pages and bound to some content to be displayed or manipulated (by *Actions*); and (ii) the navigation paths between different pages and view components, through Web links (*InteractionFlow*). Figure 2 shows an example of an IFML diagram: the Home Page ViewContainer contains a ViewComponent displaying a list of book categories. The user can click on the Selected event and thus select a category in order to see the books belonging to it in the Book List ViewComponent. By clicking on a book the usernavigates to the Book Details page, where he can see the book details.

A code generator (Figure 1 (b)) takes in input the IFML application models and deploys the web application on a web server, ready to be executed. In our implementation, we rely on the code generators of WebRatio (`www.webratio.com`) which produces Java EE applications and deploys them on any servlet container (e.g., Tomcat, as in Figure 1 (c)).

## 2.2 Analysis

The second phase (Figure 1(d)), is concerned with two main aspects: (1) the integration of the Web site log, the database containing the information displayed in the site pages, and the Web application models specified in IFML; and (2) the analysis of the resulting schema-less enriched stream. The outcome is a rich analytics able to provide deep insights regarding the behavior the user visiting the Web site. Since this is the core step of the approach, we provide full details about inputs, process and output.

**Inputs.**

Four pieces of information are consumed in this phase:

1. *The Application Model*: The IFML model of the deployed web application, as designed in the first step of the process (see Section 2.1).

2. *The Application Server Log*: The common access log produced by the server where the Web site is deployed. The access log records all the requests processed by the application server in the standard Extended Common Log Format. Listing 1 shows an example of a Web server log line.

**Listing 1: Example of Application Log line.**

```
0:0:0:0:0:0:0:1 - - [22/Jun/2016:11:10:52
    +0200] "GET /BookStore/page21.do?
kcond4.att60=11&sp=page14&link=ln67&
var6=true&var10=false&fllbck=.sv1&
cbck=wrReq32387 HTTP/1.1" 200 21973
```

3. *The Runtime Components Log*: The runtime component log (RTX) that stores events and data produced and consumed by the application runtime for serving page requests. It traces the history of the ViewContainers, ViewComponents, and operations that are executed, along with the executed queries. Moreover, the RTX log contains the data about the requested pages and their contents. Furthermore, all logged information contains a reference to the relevant application model elements and to the consumed data in the database. Each log line has the following structure (exemplified in Listing 2):

   - *Timestamp*, the time when the logged instruction is written in the log;
   - *Log Level*, the level to which the log line belongs;
   - *Host*, the host sending the request, i.e., the IP address of the client computer used by user to navigate the application;

- *Java* class, the Java class executed;
- *Session ID*, the User session identifier;
- *Model element*, the model element executed;
- *Message*, the log message.

Listing 2 shows an example of RTX log line.

**Listing 2: Example of WebRatio RTX log line.**

```
22 Jun 2016 11:10:51,761 DEBUG [http-bio
-8080-exec-5] (com.webratio.rtx.core.
ServiceProvider:45) - [119354
A67C7C0177D4A7F411E75BCDE7][page21][
pwu6Block] Creating service: WEB-INF/
descr/pwu6Block.descr
```

4. *Database*: Detailed reference to the database used to populate the requested pages.

While the application model is given at design time and we can assume it as persistent, the information contained in the logs and in the database changes as the users visit the Web site. The model can be changed in order to add or evolve features in the web application, but in that case the whole analysis process will be restarted.

**Process.**
The analyzer (Figure 1(d)) integrates the logs with the IFML model and database content. More precisely, an *enriched log* is created by merging the RTX log and the Application Server log. The enriched log combines the request of a user (recorded in the Application Server log) with the corresponding model element computed for serving the page content (i.e., the component of the page visualized). The enriched log is then merged with the IFML model of the application and the Database content. This produces a *global log*, which combines: (i) user requests, (ii) application model elements, (iii) content schema, and (iv) content instances. From this global, denormalized view one can generate any desired behaviour analysis.

**Output.**
With our approach we can compute three kinds of user behaviour analytics:

- **Navigation-based analytics:** comprehends information regarding how the users navigate the Web site (e.g., average time spent on a page, number of requests for a page, links followed, and so on). These are the typical metrics computed also by other tools.
- **Content-based analytics:** comprehends information regarding the domain entities involved in the user interaction, their types and their semantics. For example, in case of an e-commerce Web site selling in books, these metrics may include the top-$k$ visualized book, the top-$k$ clicked authors and so on.
- **Structure-based analytics:** comprehends information regarding the kind of widget, visualization, or even navigation pattern used in the user interaction model. For instance, analytics may include metrics like: top-$k$ elements clicked by users when shown in a map throughout the site, or top-$k$ elements clicked when shown in the first three positions of a list, or top-$k$ elements clicked when an attribute of type image is shown in the page versus an attribute of type currency.

Notice that each analytics result may refer to global statistics for the whole website, or to one or more model elements (e.g., *ViewCompoment*, *ViewContainer* and *InteractionFlow*).

## 2.3 Visualization

The results of the analyses can be visualized in two ways:

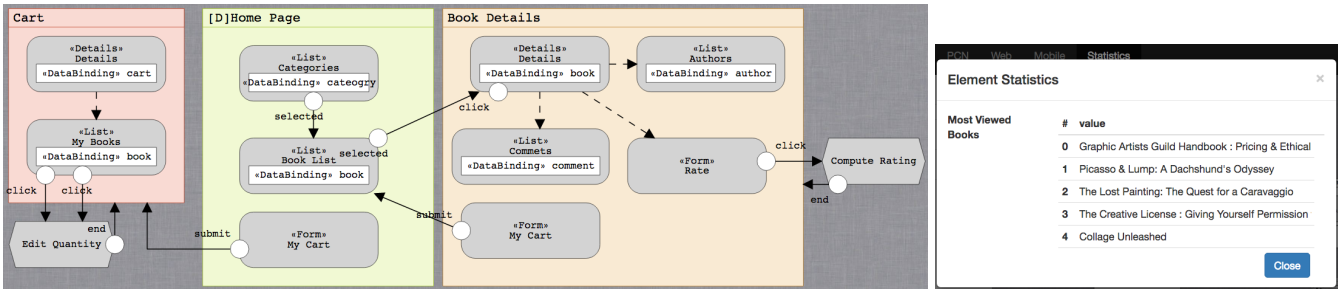
- Using a canonical data visualization tool (Figure 1(e)), which shows traditional charts like pie charts, bar charts, navigation flow charts and so on;
- Showing the data directly on the visual model (i.e., on the IFML diagrams), thus providing direct feedback of the users' behavior to the interaction designer (Figure 1(f)).

Our main contribution on the visualization side focuses on the latter; in particular we devised three means of visualization:

- **Color**: the analytics is shown through the change of color of the corresponding model element (e.g., *ViewComponent* or *ViewContainer*) based on some quantitative analysis result associated to it. This is useful for statistics that refer to many model elements and provides a immediate bird's-eye view of the user behavior for the whole website: for instance, one may color the pages in the IFML diagram based on the number of page views achieved by each of them. For instance, Figure 3(a) shows an IFML diagram with colored pages based on the number of visits per page (from green to red, where green means higher number of visits);
- **Label**: the analytics is shown with a label on the corresponding model element. For instance, one may display the number of clicks that a widget or a link has received; or the most clicked content instance(s) in a list, such as the top clicked book;
- **Properties**: the analytics is shown in a separate property panel, detailing the data about the corresponding model element. This is useful when many statistics or multi-valued results need to be displayed for the same element. For instance, Figure 3(b) shows the list of top-10 visited elements in a list.

# 3. DEMO

During the demo we will allow the visitors to experience the whole application development lifecycle as described in Section 2. For allowing users to grasp the contribution of our work, we will showcase a set of simple applications. The main case will be a small e-commerce-like web site selling books. Some excerpts of the book store application models are visible in Figures 2 and 3(a) . During the demo, the visitor will be able to look at the application models, navigate the generated applications, look immediately into the generated logs and experience the execution of the whole data process pipeline. Since the data produced and processed is large, the entire process of analysis is performed on the




(a) (b)

**Figure 3: Example of statistics displayed as colors on the IFML diagram (a) and in a property panel (b).**

Microsoft Azure platform, using an Apache Spark Cluster as processing engine and technology for the processing and computation of the required analysis. Scala was used as a programming language to write the Spark application. Visitors will be able to see the pipeline, see it at work, and look at the results in our model-based visualization approach. This will allow them to perceive the advantage of this innovative way of displaying the application analytics.

In particular, the analytics that visitors will be able to see include navigation-based ones:

- Entrance rate: the number of time a page is the starting point of the navigation of a user.
- Bounce rate: the number of time a page is the final point of the navigation of a user.
- Fixed a page, the percentage of time a link is clicked.
- Fixed a page, the percentage of time a link is used to come in that page.
- Average time spent on a page.
- Average time a page is visited.

and content-based ones:

- For each *ViewComponent* the top 10 visualized instances.
- For each *ViewComponent* of type *List* the top 10 clicked instances.

A video of the demo is available at: `http://datascience.deib.polimi.it/bigdata-modeling-weblogs`.

# 4. CONCLUSION

The main contribution of this work is to show the feasibility and advantages of a model-driven approach that blends design-time information and runtime execution data of Web sites for generating insightful analyses of user behavior.

Thanks to the proposed information fusion, new kinds of analysis can be performed, which highlight the role and meaning of visualized data in the Web pages. New analytics results can be generated based on the displayed objects, their categorizationì and their properties. Results of the analysis can be immediately displayed directly on the visual models of the applications, thus making it immediate for a designer to spot possible problems or advantages of a given user interaction design choice.

# 5. AUTHORS


## Carlo Bernaschina

Carlo Bernaschina is a PhD candidate in DEIB at Politecnico di Milano. His research interests include methods and infrastructures for Mobile and IoT development with particular focus on the application of model-driven techniques to mobile application development.

## Marco Brambilla

Marco Brambilla, Ph.D., is associate professor at Politecnico di Milano. His research covers modeling theory and methodology, Web engineering, crowdsourcing, big data and social media analytics. He is co-inventor of IFML and co-founder of Fluxedo and WebRatio.

## Thanas Koka

Thanas Koka is a MsC student at Politecnico di Milano. He is working to his thesis on big data analysis and model-driven applied to user behavior analytics.

## Andrea Mauri

Andrea Mauri, Ph.D., is a PostDoc at Politecnico di Milano. His research interests include crowdsourcing and human computation, with a particular focus on new methodologies for developing crowd- and social-based applications.

## Eric Umuhoza

Eric Umuhoza, Ph.D., is a PostDoc at Politecnico di Milano. His main research interests include domain-specific modeling and code generation for Web, mobile and IoT-based applications.


# 6. REFERENCES


[1] D. Ameller, X. Franch, C. Gómez, J. Araujo, R. B. Svensson, S. Biffl, J. Cabot, V. Cortellessa, M. Daneva, D. M. Fernández, et al. Handling non-functional requirements in model-driven development: an ongoing industrial survey. In *2015 IEEE 23rd International Requirements Engineering Conference (RE)*, pages 208–213. IEEE, 2015.

[2] D. Amyot and G. Mussbacher. User requirements notation: The first ten years, the next ten years. *Journal of Software*, 6(5):747–768, 2011.

[3] M. Brambilla and P. Fraternali. *Interaction Flow Modeling Language âĂŞ Model-Driven UI Engineering of Web and Mobile Apps with IFML*. The OMG Press. Morgan-Kaufmann, 2014.

[4] M. Brambilla, P. Fraternali, et al. IFML: Interaction Flow Modeling Language. Technical report, Object Management Group (OMG), 2014. http://www.ifml.org.

[5] M. Brambilla, A. Mauri, and E. Umuhoza. Extending the interaction flow modeling language (ifml) for model driven development of mobile applications front end. In *International Conference on Mobile Web and Information Systems*, pages 176–191. Springer, 2014.

[6] R. Breu and J. Chimiak-Opoka. *Towards Systematic Model Assessment*, pages 398–409. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.

[7] L. Cordeiro and B. Fischer. Verifying multi-threaded software using smt-based context-bounded model checking. In *Proceedings of the 33rd International Conference on Software Engineering*, ICSE '11, pages 331–340, New York, NY, USA, 2011. ACM.

[8] J. Di Rocco, D. Di Ruscio, L. Iovino, and A. Pierantonio. Mining metrics for understanding metamodel characteristics. In *Proceedings of the 6th International Workshop on Modeling in Software Engineering*, MiSE 2014, pages 55–60, New York, NY, USA, 2014. ACM.

[9] H. Giese, M. Tichy, S. Burmester, W. Schäfer, and S. Flake. Towards the compositional verification of real-time uml designs. *SIGSOFT Softw. Eng. Notes*, 28(5):38–47, Sept. 2003.

[10] J. L. Ledford, J. Teixeira, and M. E. Tyler. *Google analytics*. John Wiley and Sons, 2011.

[11] E. Umuhoza and M. Brambilla. Model driven development approaches for mobile applications: A survey. In *International Conference on Mobile Web and Information Systems*, pages 93–107, 2016.

[12] E. Umuhoza, H. Ed-douibi, M. Brambilla, J. Cabot, and A. Bongio. Automatic code generation for cross-platform, multi-device mobile apps: Some reflections from an industrial experience. In *Proceedings of the 3rd International Workshop on Mobile Development Lifecycle*, MobileDeLi 2015, pages 37–44, New York, NY, USA, 2015. ACM.

[13] M. Volter, S. Benz, C. Dietrich, B. Engelmann, M. Helander, L. C. L. Kats, E. Visser, and G. Wachsmuth. *DSL Engineering - Designing, Implementing and Using Domain-Specific Languages*. dslbook.org, 2013.