Hypersonic – Model Analysis as a Service

Vlad Acretoaie and Harald Störrle

Department of Applied Mathematics and Computer Science,
Technical University of Denmark
rvac@dtu.dk, hsto@dtu.dk

Abstract. Hypersonic is a Cloud-based tool that proposes a new approach to the deployment of model analysis facilities. It is implemented as a RESTful Web service API offering analysis features such as model clone detection. This approach allows the migration of resource intensive analysis algorithms from monolithic desktop modeling tools to a wide range of mobile and Web-based clients. As a technology demonstrator, a Web application acting as a client for the Hypersonic API has been implemented and made publicly available.

1 Introduction

The vast majority of modeling tools are currently deployed as rich client desktop applications. Some tools, such as those based on the Eclipse Modeling Framework (EMF), benefit from rich plug-in ecosystems. Others, such as the MagicDraw modeling environment [8], offer built-in access to remote resources (e. g. a version control server). However, recent years have seen the rich client architecture demonstrating its limitations, causing many areas of computing to consider more flexible alternatives. A notable example are Cloud-based architectures, which involve the deployment of computationally intensive tasks to a centralized and fully transparent shared pool of configurable computing resources. Lightweight clients access these resources via protocols such as RESTful Web services [4], giving rise to the "Software as a Service (SaaS)" paradigm. The adoption of this paradigm in the area of modeling tools is so far remarkably limited.

In this context, we have proposed Hypersonic [1], a RESTful Web service API offering model analysis facilities including clone detection, model difference computation, and model size computation. Model analysis is a particularly suitable application area for a service-oriented architecture, since analysis algorithms often demand extensive resources. The service-based approach to modeling tools is illustrated in Fig. 1: a subset of the model processing features deployed locally in a rich client architecture may instead be offered as services by several providers, while existing remote components such as model repositories remain unaffected.

This approach benefits all the involved stakeholders. For *modelers*, it supports access to scalable computational resources running the latest version of the modeling tool, as well as the option to mix-and-match services offered by different providers. For *tool providers*, it facilitates cross-device support and mitigates the distribution of unlicensed software. Furthermore, entry barriers for new

providers are lowered. For *IT administrators*, a service-based architecture brings a reduction in installation and maintenance workloads. Finally, the *Model-Based Software Development (MBSD) community* can expect improved standards compliance and an accelerated knowledge transfer between research and industry.

The comparatively few drawbacks of a service-based approach are caused by its distributed nature. Uploading large models may create a performance bottleneck, while new security and privacy aspects come into play. Nevertheless, these drawbacks are common to most Software as a Service (SaaS) solutions, and have not undermined this architecture style's acceptance.

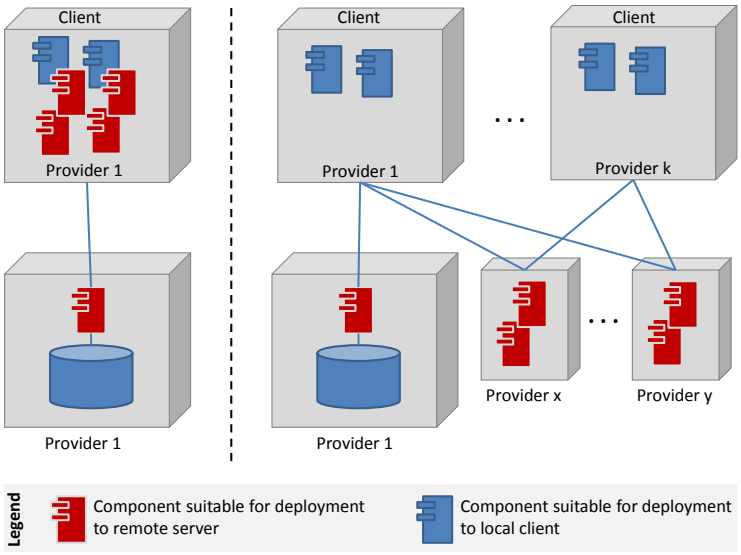

Fig. 1. Modeling tool architectures: rich client (left); Hypersonic (right)

2 Hypersonic

2.1 The Hypersonic API

The architecture of the Hypersonic API is presented in Fig. 2. Following the REST architectural style, it exposes resources for clients to interact with via HTTP requests. The API currently supports MagicDraw UML models (support for EMF and Simulink [12] models is also planned). API response messages contain the analysis results encoded as JSON [5] documents.

The Hypersonic API is implemented as a wrapper around the MACH model analysis engine [11], though from a conceptual standpoint any other model analysis engine may be used. The *RESTful API* component handles HTTP interactions with clients, delegating processing tasks to the MACH engine. All uploaded

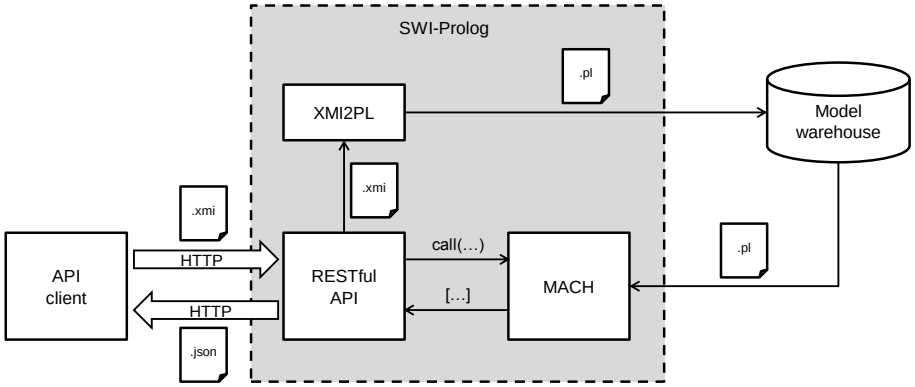

Fig. 2. Architecture of the Hypersonic API

models pass though the *XMI2PL* component, which performs a format translation from the MDXML format to the Prolog-based file format described in [10]. Once translated, models are stored in a dedicated warehouse. The *MACH* component supports operations such as clone detection, model differencing, model size computation, and basic model querying [11]. These operations can be applied to models stored in the warehouse. Analysis results are encoded by the *RESTful API* component as JSON documents to be consumed by the API client.

All processing components are executed inside a single instance of the SWI-Prolog [13] runtime. The compact representation of models as fact databases facilitates the effective in-memory processing of large models. Since some analysis operations (e. g. model size computation) are embarrassingly parallel, splitting very large models into several fact databases and executing several SWI-Prolog instances in parallel is possible, though currently not supported.

2.2 A Web-based API Client

As a technology demonstrator, a simple, mobile device friendly Web application acting as a client for the Hypersonic API[1] has been implemented. The application, written entirely in client-side JavaScript, currently supports uploading a local model to the Hypersonic model warehouse and requesting a clone detection report, which it subsequently displays in tabular form. The sample API client exemplifies our vision for Web service driven modeling tools: using Web 2.0 technologies and standards (REST, JavaScript, JSON) to enable advanced model analysis operations to be performed outside the constraints of the desktop and of traditional modeling environments.

[1] The application is available at `http://www.compute.dtu.dk/~rvac/hypersonic`.

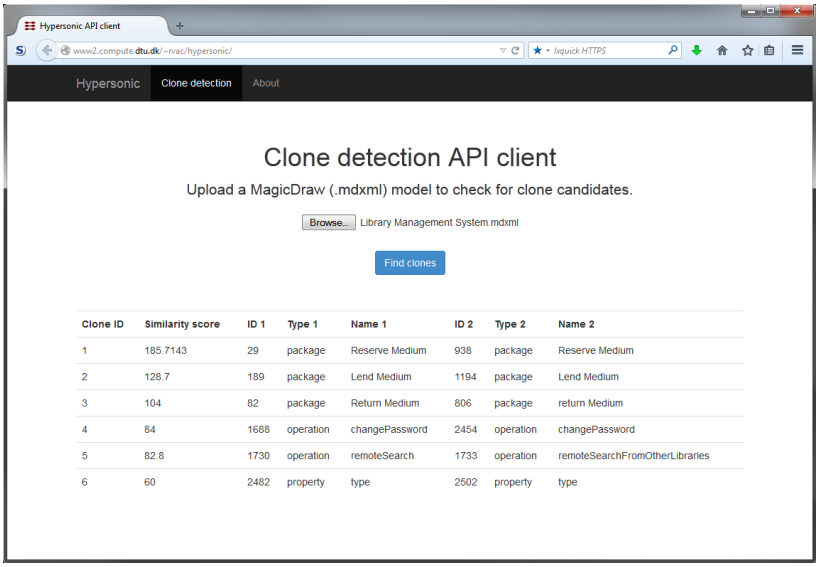

Fig. 3. Screenshot of the Hypersonic API client Web application

3 Related Work

Due to the increase in size of industrially relevant models, as well as the increase in complexity of the operations performed on them, the need for service-based, Cloud-enabled modeling tools has become apparent [7]. The main driver in this direction is the promise of performance and scalability gains for modeling activities such as model warehousing [9], querying [6], and transformation [3].

That being said, service-based model analysis has yet to receive significant attention in the literature. The closest related proposal is the EMF-REST project [2], aimed at automatically generating RESTful Web service interfaces for EMF models, much like existing EMF tools generate Java APIs for such models. However, while it does provide basic model manipulation operations, EMF-REST is not primarily designed as a model analysis tool.

4 Ongoing Developments

The Hypersonic API is undergoing development with the aim of reaching feature parity with the MACH command-line model analysis tool. Once this is achieved, the API will be deployed to a private Cloud platform. We plan to eventually extend the API operations beyond model analysis, into the realm of model querying and model transformation. In parallel, we will work towards creating an ecosystem of API clients, including smartphone/tablet apps and modeling tool plug-ins (see Fig. 4). For these proposals to become practical solutions, security aspects such as user authentication and model access control must be addressed.

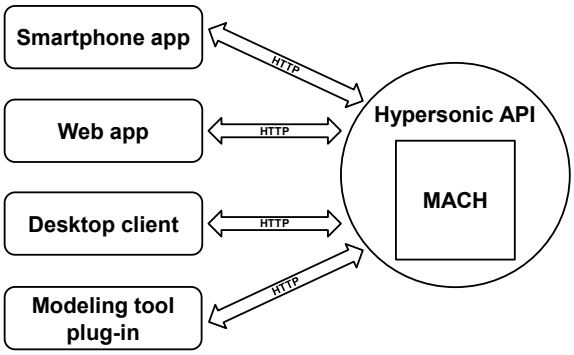

Fig. 4. Various clients consuming the Hypersonic API

References

1. Vlad Acretoaie and Harald Störrle. Hypersonic: Model Analysis and Checking in the Cloud. In *Proc. 2nd Ws. Scalability in Model Driven Engineering (BigMDE'14)*, volume 1206 of *CEUR-WS*, pages 6–13, 2014.
2. Jordi Cabot. EMF-REST. `http://emf-rest.com`, retrieved 26.08.2014.
3. Caue Clasen, Marcos Didonet Del Fabro, and Massimo Tisi. Transforming Very Large Models in the Cloud: a Research Roadmap. In *Proc. First Intl. Ws. Model-Driven Engineering on and for the Cloud (CloudMDE'12)*, pages 3–12, 2012.
4. Roy Thomas Fielding. *Architectural Styles and the Design of Network-based Software Architectures*. PhD thesis, University of California, Irvine, 2000.
5. Internet Engineering Task Force (IETF). IETF RFC 7159: The JavaScript Object Notation (JSON) Data Interchange Format, 2014.
6. Benedek Izsó, Gábor Szárnyas, István Ráth, and Dániel Varró. IncQuery-D: Incremental Graph Search in the Cloud. In *Proc. 1st Ws. Scalability in Model Driven Engineering (BigMDE'13)*, pages 4:1–4:4, New York, NY, USA, 2013. ACM.
7. Dimitrios S. Kolovos, Louis M. Rose, Nicholas Matragkas, Richard F. Paige, Esther Guerra, Jesús Sánchez Cuadrado, Juan De Lara, István Ráth, Dániel Varró, Massimo Tisi, and Jordi Cabot. A Research Roadmap Towards Achieving Scalability in Model Driven Engineering. In *Proc. 1st Ws. Scalability in Model Driven Engineering (BigMDE'13)*, pages 2:1–2:10, New York, NY, USA, 2013. ACM.
8. NoMagic, Inc. MagicDraw UML 17.0.3. `http://www.nomagic.com/products/magicdraw`, retrieved 26.08.2014.
9. Javier Espinazo Pagán, Jesúss Sánchez Cuadrado, and Jesús García Molina. Morsa: A Scalable Approach for Persisting and Accessing Large Models. In *Proc. 14th Intl. Conf. Model Driven Engineering Languages and Systems (MODELS'11)*, volume 6981 of *LNCS*, pages 77–92. Springer Berlin Heidelberg, 2011.
10. Harald Störrle. Towards Clone Detection in UML Domain Models. *J. Softw. Syst. Model.*, 12(2), 2013.
11. Harald Störrle. UML Model Analysis and Checking with MACH. In *Proc. 4th Intl. Ws. Academic Softw. Development Tools and Techniques (WASDETT'13)*, 2013.
12. The MathWorks, Inc. Simulink. `http://www.mathworks.se/products/simulink/`, retrieved 26.08.2014.
13. Jan Wielemaker, Tom Schrijvers, Markus Triska, and Torbjörn Lager. SWI-Prolog. *Theory and Practice of Logic Programming*, 12(1-2):67–96, 2012.