Abstract—In this paper we introduce vFlow - A framework for rapid designing of batch processing applications for Cloud Computing environment. vFlow batch processing system extracts tasks from the vPlans diagrams, systematically captures the dynamics in batch application management tasks, and translates them to Cloud environment API, named vDocuments, that can be used to execute batch processing applications. vDocuments do not only enable the complete execution of low-level configuration management tasks, but also allow the construction of more sophisticated tasks, while imposing additional reasoning logic to realize batch application management objectives in Cloud environments. We present the design of the vFlow framework and illustrate its utility by presenting the implementation of several sophisticated operational tasks.

I. INTRODUCTION

The rapid growth of demand on modern resource-intensive distributed enterprise applications in virtualized environments has led to a significant need to advanced tools that are capable of automating the management of such applications. Even though the research community (e.g. [5]) and industry (e.g. [3]) have been very actively exploring methods for automating resource management in grid and batch processing environments, these works mainly focus on the backend part, and consequently, there is a little work done to allow the automatic creation of batch processing applications in Cloud environments, which can substantially reduce the human factor errors and drastically improve the dynamically automated resource management process.

In this paper we present our novel state-of-the-art framework, vFlow, for rapidly building automated batch applications in virtualized environments based on Petri Net model [4]. vFlow allows Cloud users to automate application execution and management process by building distributed batch processing application flow plans known as vPlans. vPlans can be rapidly created through vFlow’s GUI-based editor, and also can be modified through the textual editor. The editor systematically captures the dynamics in distributed data-intensive complex batch processing application management tasks, validates them and translates them to formal batch application execution plan that can be executed in Cloud environments such as Amazon EC2 and GoGrid1 Our tool supports user interactions with large-scale Cloud environments through its interfaces without the need for plug-ins or special-purpose runtime systems and generates validated ready-to-run batch processing application execution plans.

II. PRESENTATION

A. Background

Petri Nets [4] are flow-chart-like diagrams that were first introduced by Carl Adam Petri in 1962 as a formal method for describing the concurrency and synchronization present in distributed systems and to determine the correctness of these systems. The main elements of Petri Nets are as follows:

- **Places**: These circular nodes represent conditions or objects such as program variables. A place may contain none, one, or multiple Tokens;
- **Tokens**: These Black dots represent the specific state or the condition or object. The places that have tokens are represent the current state of the described system;
- **Transitions**: Transitions, which are represented as rectangles, model system activities which result in a change in the current system status. Transitions can be method calls, for instance. A transition executes if all Places that have input Arcs to the transition have tokens;
- **Arcs**: These arrows connect between pairs of places and transitions. Arcs show the relative order and dependency of between various places and transitions and thus indicate which objects and conditions are changed by which activity.

For more information about Petri Nets and its usages in representing application flow, we refer the reader to [4].

B. Framework

The framework consists of the main five loosely coupled components: the vPlan GUI Editor; the vPlan Textual Editor; the vPlan Validator; the Repository (Execution vPlan Library); and the API Generator.

- **vPlan GUI Editor**: The editor is a central component in vFlow framework. The editor is the GUI-based user interface component that allows the user to create a distributed batch application flow model process based on Petri Net model techniques. The user interface is similar
to a flow chart where the user could put circles, squares and triangles, then connect them together. Although this seems simple, the diagrams can become quite complex based on the Petri Net model controls of these symbols. The editor also allows building complex hierarchical application execution plans. Within each circle in the vPlan, code can be running from Java or Ruby sources, or the user can choose to generate application execution plan document (Web Service API) by integrating the execution plan with a specific Cloud vendor’s API. The editor captures advanced features such as annotations and interrupts when an event occurs in the application running in Cloud.

- **vPlan Textual Editor**: The textual editor allows the user to modify the initial application execution plan generated by GUI Editor. This includes for instance adding comments to the plan document. The modifications made to the plan document through the Textual Editor are reflected in real-time to the GUI editor, and vice versa.

- **vPlan Validator**: This component is responsible of the verification and validation of the vPlan structure created by the user. It mainly verifies the following features: (1) **Well-Formedness**: There is exactly one main vPlan per application, which is connected, and has one start and one end places; (2) **Reachability**: There must be at least one path from the start place to the end place passing through every other place and transition in each vPlan; and (3) **Well-Structureness**: If the vPlan is hierarchical, then each lower level vPlan module referred by meta-place in the higher level vPlan is designed and exists. Furthermore, it assures that annotations represent valid arguments with respect to the corresponding transitions.

- **Repository**: This is the code repository to store all modules and related scripts. Each transition represents a call to a module or script from the repository and repository content can be reused.

- **API Generator**: Generates the final batch application execution plan in form of Web Service API. Compared to traditional scripts that at best automate the generation and modification of configuration on target application or device, a vPlan API additionally encapsulates the logical reasoning that guides the workflow of the application management task. This capability enables full automation of batch processing application tasks, minimizing human involvement.

### III. vFlow Demonstration

The demo software is a complete implementation of the vFlow framework, featuring the components described in Section II-B. In our demo we plan on showing various abilities of vFlow. The user first creates a batch application execution plan using the GUI editor and assigns each transition with one of the functions (i.e. tasks) written in advance and stored in the repository. The users alternatively can develop their own functions using Java or Ruby. Once the vPlan is validated, the API Generator generates ready-to-run vDocument (i.e. Amazon EC2 Web Service API), and the editor allows the user to go through, step-by-step, to test/debug the application or just a specific module. The user can also edit the plan via the vPlan Textual Editor. The developed vPlan can be run repeatedly based on specific configurations (e.g. once every 24 hours at 12:00 AM) and it runs in the background, so that it doesn’t affect the performance of other critical applications.

### REFERENCES

[1] D. Auber, Tulip. In P. Mutzel, M. Junger, and S. Leipert, editors, 9th Symp. Graph. Drawing, volume 2265 of Lecture Notes in Computer Science, pages 335337. Springer-Verlag, 2001.

[2] K. A. Ahmat, H. Gobjuka. Managing Clouds in Cloud Platforms, In Proc. USENIX NSDI 2010 Poster/Demo Session, 2010.

[3] J. Pathak, J. Treadwell, R. Kumar, P. Vitale, F. Fraticelli, A Framework for Dynamic Resource Management on the Grid, HP Laboratories Palo Alto, Technical Report HPL-2005-153, 2005.

[4] W.M.P. van der Aalst, The application of petri nets to workflow management, 1998.

[5] D. Warneke and O. Kao, Exploiting Dynamic Resource Allocation for Efficient Parallel Data Processing in the Cloud, In Proc. IEEE Transactions on Parallel and Distributed Systems (TPDS), 2011.