X10 を用いた分散エージェントシミュレーション基盤サービス

村田 浩樹　水田 秀行　鈴村 豊太郎　竹内 幹雄　河内谷 清久仁

我々が分散プログラミング言語 X10 上で開発した分散エージェントシミュレーション基盤（以下 *XSIM*）は，エージェントの配置，エージェント間のメッセージ交換，ノード内，ノード間のエージェントの移動，ロギングなどの機能を持つ．クラウドの一形態である PaaS(Platform as a Service) は，ソフトウェアをサービスとして開発する環境を提供するが，*XSIM* をサービスとして提供することでアプリケーション開発者の負担を軽減できる．本稿では *XSIM* サービスの設計と，その性能，スケーラビリティについて述べる．

1 はじめに

クラウド・コンピューティングが計算資源を調達する方法として一般的になっている．サービス・プロバイダと人を介したやり取りをせずに，ネットワーク，サーバ，ストレージ，アプリケーション，サービスといった資源を調達でき，利用分の費用しか発生せず，標準的なネットワーク・プロトコルで PC やモバイルデバイスからアクセスできるといった利点がある．クラウド・コンピューティングには 3 つのサービスモデルがある．Software as a Service (SaaS), Platform as a Service (PaaS), Infrastructure as a Service (IaaS) である [4]．SaaS はサービス・プロバイダがアプリケーションをサーバ込みでネットワーク経由でユーザに提供する．PaaS はユーザにアプリケーションを開発するための基盤とサーバを提供する．IaaS はソフトウェアを配置するための計算資源を提供するというモデルである．

分散プログラミング言語 X10 [1] は並行処理や分散処理，局所性を柔軟に扱うために開発されており，APGAS と呼ばれるモデルを用いている．このモデルではグローバル・アドレス空間を複数のプレースという領域に分割する．X10 はこのプレースを同期した共有メモリ計算コンテクストとして導入している．各プレースはデータを持ち 1 つ以上のタスク（X10 ではアクティビティと呼ぶ）を実行する．タスクは `async` 文を用いてタスクを作り出すことができ，`at` 文を用いて他のプレースに移動し，そこにあるデータをアクセスすることができる．また，X10 プログラムは Java™ または C++ に変換され，コンパイルされる．Java に変換するものを Managed X10，C++ に変換するものを Native X10 と呼ぶ．

XSIM は X10 を用いた社会シミュレーション用の分散エージェント・シミュレーション基盤である [5] [6]．アプリケーションは Java で書かれるため，Java の API を持っている．エージェントの配置，エージェント間のメッセージ交換，ノード内，ノード間のエージェントの移動，ロギングなどの機能を持ち，エージェント・シミュレーションを多数のノード上で分散実行できる．*XSIM* を PaaS の開発基盤の一部であるサービスとして提供することで，開発環境の構築や

X10-Based Distributed Agent Simulation Platform as a Service.

Hiroki Murata, Hideyuki Mizuta,
日本アイ・ビー・エム（株）　東京基礎研究所
国立研究開発法人科学技術振興機構，CREST,
IBM Research - Tokyo
CREST, JST.

Toyotaro Suzumura, IBM T.J. ワトソン研究所, IBM T.J. Watson Research Center.

Mikio Takeuchi, Kiyokuni Kawachiya, 日本アイ・ビー・エム（株）　東京基礎研究所, IBM Research - Tokyo.

環境の拡張，縮小が容易になり，アプリケーション開発者はシミュレーション自体の開発に集中できるようになる．またシミュレーションの開発時や通常時は少ない数のノードで実行しておいて，例えば，災害発生時に避難シミュレーションを速やかに実行するために同じ環境で簡単にノード数を増やして高速なシミュレーションをするといったことができる．本稿は，この「*XSIM* サービス」の設計とその性能，スケーラビリティについて述べる．

2 XSIM: X10 による分散エージェントシミュレーション基盤

XSIM のエージェントプログラミングモデルは我々が以前開発した Java によるエージェントシミュレーション基盤（以下 JSIM）[10] のものを用いている．*XSIM* は Java の API を持ち，エージェントの配置，エージェント間のメッセージ交換，ノード内，ノード間のエージェントの移動，ロギングなどの機能を持つ．*XSIM* は Java と X10 を用いて実装されており，Managed X10 の Java interoperability [7] が使われている．Java で実装されている部分は主に Java の API であり JSIM の実装を利用している．X10 で実装されている部分はエージェント間のメッセージ交換とエージェントの移動であり，分散計算機での明示的な通信を使わず，X10 の `at` 文を用いて他の プレースのメソッドを呼ぶ形で実装されている．図 1 に 4 ノードに配置された *XSIM* のアーキテクチャの例を示す．これらのノードには X10 の実行環境が構築されており，各ノードには 1 つのプレースが配置され他のノードとは証明書を用いたパスワードなしの ssh で接続できる．ユーザのシミュレーションは各ノードに配置され，*XSIM* によってロード，実行される．図 1 の中の “UI” はシミュレーションで使用するデータのアップロード，結果のダウンロード，GUI の提供などを行う．*XSIM* 環境はエージェントを分散実行でき高速なシミュレーションを行うことができるが，複数の計算機を用意し X10 と *XSIM* の環境をセットアップすることは単にシミュレーションを行いたいユーザにはしきいが高い場合がある．そこで本稿では *XSIM* 環境をクラウドを通じて提供する方法を検討する．

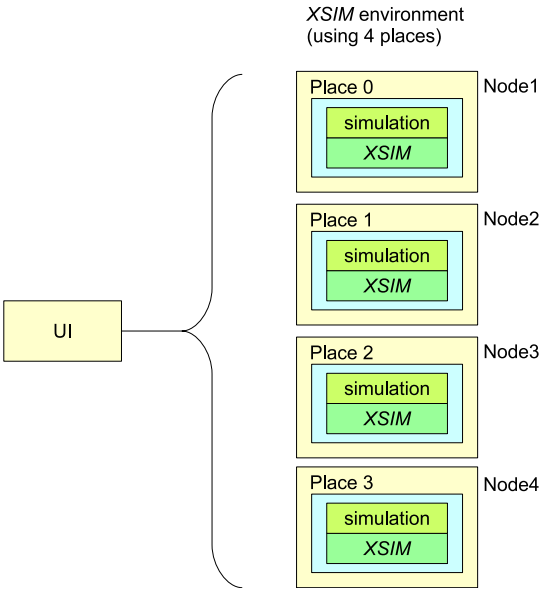

図 1 *XSIM* **アーキテクチャ**

3 設計

XSIM 環境をクラウドを通じて提供する場合，*XSIM* 環境のセットアップされたノードを提供して，ユーザには “UI” とシミュレーション・プログラム (“simulation”) を用意，配置してもらうようにすると，PaaS 上に図 1 をそのまま再現した形になる．しかし，大規模なシミュレーションを行うためにシミュレーション用のノードを増やしていくと，シミュレーション・プログラムやシミュレーション用のデータの配置や結果の回収が難しくなる．それらを自動化するようなサービスを用意することでユーザのシミュレーション実行を容易にすることができる．

図 2 は，そのようなサービスを追加して構築した *XSIM* 環境を示している．図中の “UI” と “VM1–4” が図 1 の “UI” と “Node1–4” に相当し，“*XSIM* Service” が追加したサービスである．“UI” はシミュレーション用データのアップロードや結果のダウンロード，GUI の提供を行うので通常ウェブアプリケーションとして実装される．ウェブアプリケーションは PaaS 上での実装が容易なので PaaS 上に実装する．“Node1–4” は ssh アクセスする必要があるが，PaaS

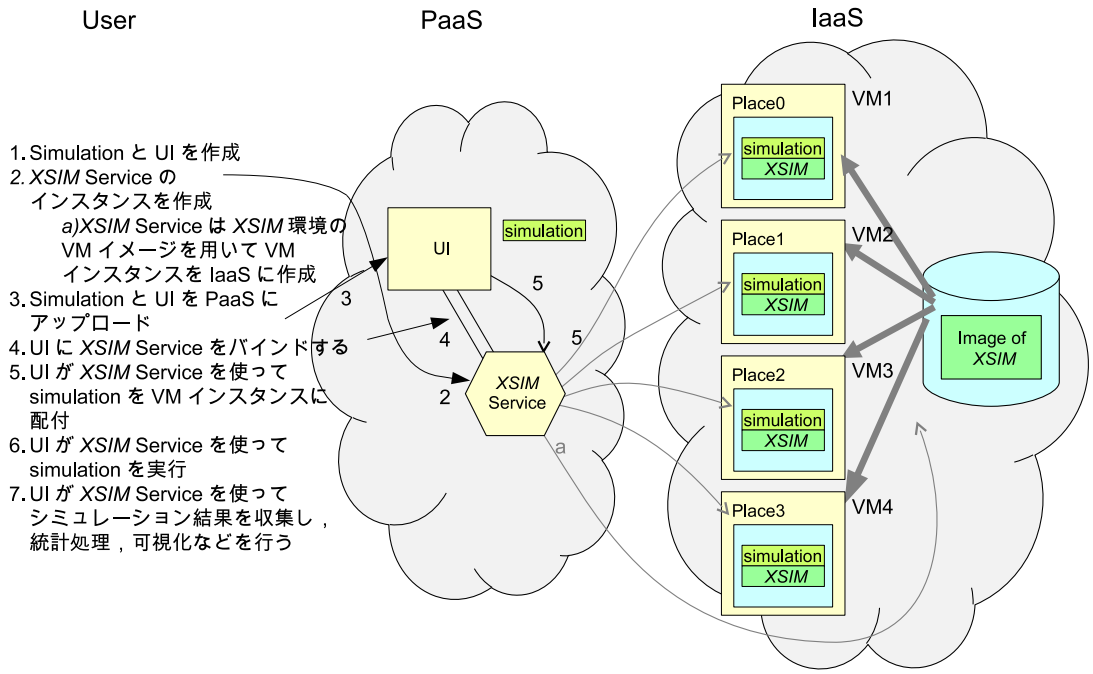

図 2　*XSIM* サービスのアーキテクチャ

表 1　*XSIM* Service API

API	Route
provision	PUT /service_instances/:instance_id
bind	PUT /service_instances/:instance_id/service_bindings/:binding_id
unbind	DELETE /service_instances/:instance_id/service_bindings/:binding_id
deprovision	DELETE /service_instances/:instance_id
upload	POST /instance_id2/upload
execute	PUT /instance_id2/execute/:file_spec
download	GET /instance_id2/download/:file_spec
change VM number	PUT /instance_id2/change_vm_number/:VM number

では ssh アクセスできない場合が多いため，IaaS 上に配置している．“*XSIM* Service” は，PaaS のサービスとして必要な REST API と，*XSIM* 環境を管理するための API を兼ね備えた Web サービスである．REST API は PaaS のサービスとして登録するために必要な，インスタンスの生成 (provision)，削除 (deprovision)，アプリケーションへのバインド (bind)，アンバインド (unbind) といった API と，ユーザのシミュレーション・プログラムとデータの “VM” への配置 (upload)，シミュレーションの実行 (execute)，結果の回収 (download)，VM 数の増減 (change VM number) といった VM に配置した *XSIM* 環境を管理するための API からなる．以下，API の説明のため，PaaS として Cloud Foundry [2] を想定する．PaaS のサービスはそれぞれ固有の url，user name，password で登録されており，url に各 API の引数を加えてアクセスする．表 1 に API の例を示す．以下，各 API について説明していく．

3.1 provision

provision の際には PaaS システムが instance id を割り当て，それを組み込んだ表 1 の Route を url の後につなげてアクセスする．*XSIM* Service は，IaaS 上に *XSIM* 環境の VM イメージを用いて VM を provision するか，すでに provision されている VM を instance id と関連付けて DB に登録する．

3.2 bind

バインドの際には PaaS システムが binding id を割り当て，instance id と共に表 1 の Route のように組み上げて url の後につなげてアクセスする．*XSIM* Service は，binding id を instance id と関連付けて登録する．instance id2，user id2，password2 と証明書を生成，登録し，instance id2，user id2，password2 と証明書の公開鍵をシステムに返す．システムは，ユーザ・アプリケーション (“UI”) の環境変数に instance id2 を後につなげた url と，user id2，password2，公開鍵を設定し，ユーザ・アプリケーションから *XSIM* Service にアクセスできるようにする．

3.3 unbind

アプリケーションにバインドした *XSIM* Service が不要になった場合は，アンバインドして削除（deprovision）する．アンバインドの際には PaaS システムが，instance id と binding id を表 1 の Route のように組み上げて url の後につなげてアクセスする．*XSIM* Service は，binding id と，関連付けられた instance id2，user id2，password2，証明書の登録と VM 上のデータを抹消する．

3.4 deprovision

deprovision の際には PaaS システムが，instance id を表 1 の Route のように組み上げて url の後につなげてアクセスする．*XSIM* Service は，instance id と関連付けられた VM を deprovision して登録を抹消するか，証明書を削除した後，空きの VM として登録し直し，instance id の登録を抹消する．

3.5 upload

ユーザのシミュレーション・プログラムとデータの “VM” への配置は，*XSIM* Service を “UI” にバインドした後，“UI” から *XSIM* Service にシミュレーション・プログラムとデータを zip して REST API を用いてアップロードすることで行う．必要であれば VM の公開鍵で暗号化してアップロードする．*XSIM* Service はシミュレーション・プログラムとデータを受け取ると，1 つの VM に送り，そこから登録されている他の VM に配付し，暗号化，zip を解く．

3.6 execute

シミュレーション・プログラムは，*XSIM* によってロード，実行される．実行するプログラムは *XSIM* の設定ファイルで指定されるので，ユーザは実行するプログラム指定した設定ファイルをデータと共にアップロードする．シミュレーションの実行だけであれば特定のコマンドを実行すればよいが，実行のモニタなどのために指定したコマンドを実行できるような REST API を用意する．

3.7 download

シミュレーション結果の収集は，指定したファイルを収集し 1 つの VM にディレクトリごとに分けて置き zip して，*XSIM* Service を経由して “UI” に戻す．必要であれば各 VM で暗号化した後，収集する．

3.8 change VM number

VM 数の増減は，“UI” から VM 数を指定して *XSIM* Service の REST API を呼び出すことで行う．VM を増やす場合には，*XSIM* Service は IaaS 上に *XSIM* を利用できる VM イメージを用いて VM を provision するか，すでに provision されている VM を instance id と関連付けて DB に登録する．証明書を生成し，登録した VM に配布してパスワードなしの ssh で接続できるようにする．また，シミュレーション・プログラムとデータを配付する．VM を減らす場合には，*XSIM* Service は VM を deprovision して DB の登録を抹消するか，空きの VM として登録しなおす．

図 3　Bluemix カタログの *XSIM* サービス

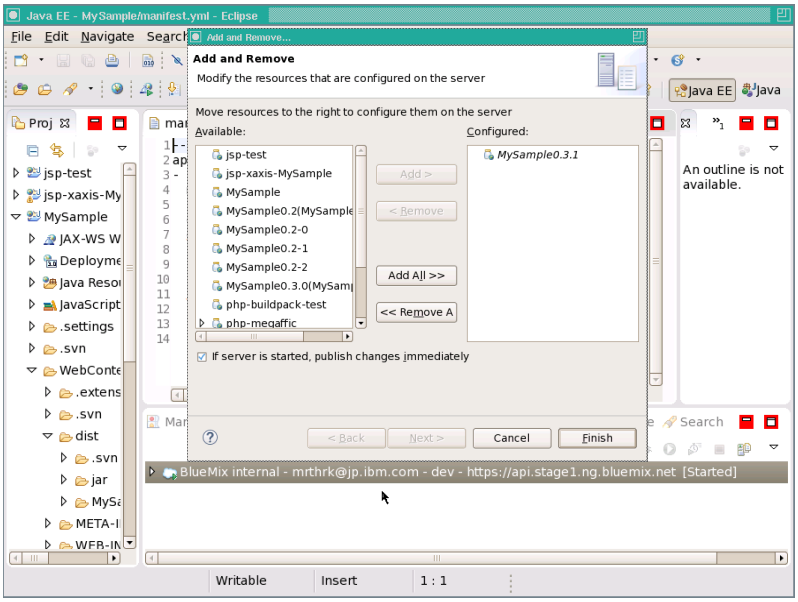

図 4　Eclipse からプロジェクトをアップロードする

3.9　ユースケース

図 2 の左側の “User” の列は *XSIM* サービスの使い方を示している．

1. ユーザはシミュレーション・プログラム (“simulation”) とそのための UI プログラム (“UI”) を Dynamic Web Project として Eclipse 上で作成する
2. *XSIM* Service のインスタンスを作成．図 3 は Cloud Foundry をベースとした IBM の提供する PaaS である Bluemix ™ [3] 内で，作成するサービスを選択する画面で *XSIM* Service のアイコンが 1 行目の 2 番目に表示されている．

3. シミュレーション・プログラムとシミュレーション用データ，UI プログラムを PaaS にアップロードする．図 4 は，下のペインに登録されているサーバにアップロードするプロジェクトを選択している画面である．
4. UI プログラムの設定ファイルに *XSIM* Service をバインドすることを記述しておくとアップロード後に自動でバインドされる
5. UI プログラムが *XSIM* Service を使ってシミュレーション・プログラムとシミュレーション用データを VM インスタンスに配付する
6. UI プログラムが *XSIM* Service を使ってシミュレーション・プログラムを実行
7. UI プログラムが *XSIM* Service を使ってシミュレーション結果を収集し，統計処理，可視化などを行う．

このような構成にすることでユーザは Eclipse 上で XSIM 上で動作するシミュレーション・プログラムと GUI を用意するだけで，環境などのセットアップをすることなく容易にシミュレーションを実行することが可能になる．

4 評価

本節では完全グラフでのメッセージ交換を行うエージェント・アプリケーションを使って *XSIM* サービスを評価する．実験環境は Bluemix 上に構成した *XSIM* Service と IBM の提供する IaaS である SoftLayer® [3] 上に構成した 1–4 台の VM からなり，各 VM は x86 2GHz 8 コアと 16GB メモリを持つ．OS は CentOS 6.x の 64bit 最小インストール．X10 2.5.0 (Managed X10）と IBM®Java 1.7.0 を用意した．図 5 は総エージェント数を 256 とした場合のアプリケーションの strong scale の性能を示す．値は 3 回の実行の平均である．アプリケーションの実行時間はプレース数の増加にともない減少するが，線形には減少していない．1，2，4 プレースは，それぞれ 1，2，4 台の VM で実行されているが，8，16 プレースは 4 台の VM で実行されている．アプリケーションは 2 スレッドで実行されるが計算と通信はシリアルに実行される．アプリケーションの実行には 1 プレースで 1.6 秒，2 プレースで 1.2 秒かかっている．1 プレースは 1 つの VM 上で実行されているため計算にほぼすべての時間がかかっていると考えられる．この計算を 1 つの VM 上の 2 プレースで実行すると半分の 0.8 秒で済むはずであるが，実験では 2 プレースは 2 つの VM 上で実行されており通信に 0.4 秒かかっていることになる．4 プレースになると通信にかかる時間は 1 プレースあたりのエージェントが 1/2 になるが別の VM の相手エージェントが 1.5 倍になるため 3/4 になり，計算時間が 1/2 になるため，実行時間はおよそ 0.7 秒と計算できる．これ以上プレースを増やしても VM が増えないため通信にかかる時間は減らず，一方プレースが増えると計算時間が 1/2 になっていき，実行時間は 8 プレースでおよそ 0.5 秒，16 プレースでおよそ 0.4 秒と計算できる．これは実験結果とほぼ一致している．このように必要な性能に応じて VM 数を増減し性能をスケールさせることができた．

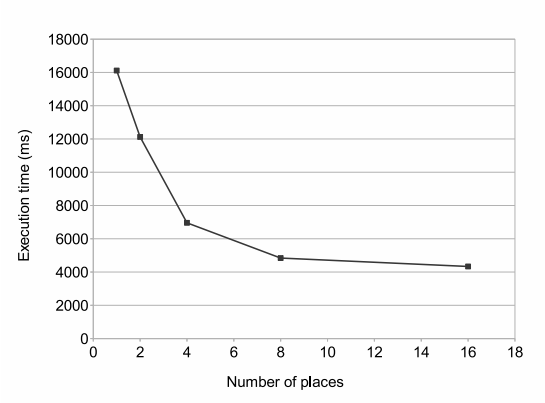

図 5　完全グラフでのメッセージ交換を行うエージェント・アプリケーションの性能

5 関連研究

クラウド・コンピューティング・サービス上にエージェント・シミュレーションを用いた社会シミュレーション・システムを構築したものとしては Wittek らの [9] があげられる．Wittek らは，シミュレーション・ソフトウェアをクラウド・コンピューティング・サービス上に移植して実行し，メッセージング性能を評価しているが，サービス化は行っていない．サービ

ス化を行ったものとしては，Taylor らの [8] や Zehe らの [11] などがあげられる．Tylor らはクラウド・コンピューティング・サービスを対象としたシミュレータ向けワークフロー・マネージャを用いてシミュレーション・ソフトウェアをクラウド・コンピューティング・サービス上で動かすケース・スタディを行っている．Zehe らは，シミュレーション・ソフトウェアを IaaS 上に構築し SaaS として提供する際のアーキテクチャ，特にデータのセキュリティについて検討し，実装，評価しているが，いずれもシミュレーション・プラットフォームのサービス化は行っていない．

6 まとめ

本稿では X10 上で開発した分散エージェントシミュレーション基盤 (*XSIM*) を PaaS と IaaS を組み合わせた環境にサービスとして設置する際の設計と *XSIM* のサービスとしての利用方法について述べた．完全グラフでのメッセージ交換を行うエージェント・アプリケーションを使って *XSIM* サービスの性能を測定し，そのスケーラビリティを確認した．今後は *XSIM* サービスの自動スケーリングへの対応，*XSIM* 上のエージェント・シミュレーション・アプリケーションの提供を進める予定である．

参 考 文 献

[1] Charles, P., Grothoff, C., Saraswat, V., Donawa, C., Kielstra, A., Ebcioglu, K., Praun, C. V., and Sarkar, V.: X10: an object-oriented approach to non-uniform cluster computing, *Proceedings of the 20th annual ACM SIGPLAN conference on Object-oriented Programming, Systems, Languages, and Applications*, 2005, pp. 519–538.

[2] Cloud Foundry Foundation: Cloud Foundry, `https://www.cloudfoundry.org/`.

[3] International Business Machines Corp.: IBM Cloud, `http://www.ibm.com/cloud-computing/`.

[4] Mell, P. M. and Grance, T.: SP 800-145. The NIST Definition of Cloud Computing, Technical report, National Institute of Standards and Technology, 2011.

[5] Suzumura, T. and Kanezashi, H.: Highly Scalable X10-Based Agent Simulation Platform and Its Application to Large-Scale Traffic Simulation, *Proceedings of the 2012 IEEE/ACM 16th International Symposium of Distributed Simulation and Real Time Applications*, 2012, pp. 243–250.

[6] Suzumura, T., Kato, S., Imamichi, T., Takeuchi, M., Kanezashi, H., Ide, T., and Onodera, T.: X10-based massive parallel large-scale traffic flow simulation, *Proceedings of the 2012 ACM SIGPLAN X10 Workshop*, 2012.

[7] Takeuchi, M., Cunningham, D., Grove, D., and Saraswat, V.: Java Interoperability in Managed X10, *Proceedings of the Third ACM SIGPLAN X10 Workshop*, X10 '13, 2013, pp. 39–46.

[8] Taylor, S. J. E., Anagnostou, A., Kiss, T., Terstyanszky, G., Kacsuk, P., and Fantini, N.: A Tutorial on Cloud Computing for Agent-based Modeling & Simulation with Repast, *Proceedings of the 2014 Winter Simulation Conference*, WSC '14, 2014, pp. 192–206.

[9] Wittek, P. and Rubio-Campillo, X.: Scalable agent-based modelling with cloud HPC resources for social simulations, *Cloud Computing Technology and Science (CloudCom), 2012 IEEE 4th International Conference on*, 2012, pp. 355–362.

[10] Yamamoto, G., Tai, H., and Mizuta, H.: A Platform for Massive Agent-based Simulation and its Evaluation, *Proceedings of the Sixth Intl. Joint Conf. on Autonomous Agents and Multiagent Systems*, 2007.

[11] Zehe, D., Knoll, A., Cai, W., and Aydt, H.: SEMSim Cloud Service: Large-scale urban systems simulation in the cloud, *Simulat. Modell. Pract. Theory*, (2015).