Global Headquarters: 5 Speen Street Framingham, MA 01701 USA P.508.872.8200 F.508.935.4015 www.idc.com

TECHNOLOGY ASSESSMENT

Java：買収から 2 年半後における状況

Al Hilwa
(英語原文: 2012 年 8 月発行)

IDC の見解

オラクルが 201 年 1 月にサン・マイクロシステムズの買収を完了してから 2 年半が経過したが、Java コミュニティで広く懸念されていたことの多くは現実のものとなっていない。オラクルは大半の意思決定に際して周到で断固たるアプローチを取っており、これは Java の長期的な展望に対するコミュニティの信頼感を高めるものであると見られる。これに加えて、以下の事項が特筆される。

- オラクルは、Apache ソフトウェア財団が解除を主張していた認定製品の使用に係る制限を認めず、Apache Harmony の Java 実装の承認に係る長期に渡る紛争を断固とした態度で終結させた。
- オラクルは IBM、アップル、SAP などの主要ベンダーを Java のオープンソース JDK である OpenJDK のメンバーとして迎え入れ、OpenJDK を Java テクノロジーのリファレンス実装として選定した。
- オラクルは買収以前に計画されていた改善を 2 つのリリースに分割し、Java Standard Edition（SE）7 の出荷を問題なく完了した。結果的に、Java は Sun 買収後の 2 年半において、買収前の 2 年半と比べて大幅に前進している。
- オラクルは、Java Community Process（JCP）を通じて、Java SE 8 および Java Enterprise Edition（EE）7 に対するより現実的なロードマップを提示している。この 2 つの新しいリリースを予定通り提供することが、オラクルの Java 戦略の妥当性に対する重要な検証となる。
- Java のエコシステムは堅調であり、依然として成長軌道にある。かつてない数のプログラミング言語が Java 仮想マシン（JVM）上にホスティングされており、多くの新規開発者がモバイル Android 開発者として、広範な Java 開発者のエコシステムをさらに強化している。
- Java は競合する開発者エコシステムからの圧力の下にある。競合するエコシステムには積極的に運営されているマイクロソフトプラットフォームのエコシステムや、多様なテクノロジーと軽量のスクリプト言語およびフレームワークを備えた広範な Web エコシステムなどが含まれる。新規開発者に対する魅力を維持するためには、Java はより早いスケジュールで進化し、業界で現在進行中のモバイルやクラウド、ソーシャルアプリケーションへの転換を効果的にサポートする必要がある。
- Java には、グーグルの Android での成功と、Android のクライアントおよびサーバーのフォームファクターへの進化の可能性によって、類似しているが相互に競合する複数のテクノロジーに細分化するリスクがある。

©2012 IDC

調査概要

オラクルは 2009 年 4 月のサン・マイクロシステムズ（以下、サン）の買収発表に伴って実施されたさまざまな規制当局による 2 年間の審査の後、2010 年 1 月 27 日に買収の完了を発表した。この買収により Java テクノロジーがオラクルの管理下に入ったことは、Java コミュニティではオラクルによる Java の処遇に関して不安と困惑をもたらした。本調査では、オラクルの下における Java の状況に関して詳細な検討を行う。

概況

2008 年後半に IBM はサンに買収の可能性を打診し、内部での数か月に渡る議論が始まった。これは 2009 年 3 月 18 日にニュースとなった。サンはこの議論の過程で他社からの買収提案を求め、2009 年 4 月 20 日にオラクルによるサンの買収で合意に至ったと両社が共同で発表した。米国と欧州においては、この買収によって発生する競争上の問題を巡って長期に渡る規制審査のプロセスが開始された。この期間において、業界やオープンソースコミュニティではこの買収が Java の将来に及ぼす影響に関するさまざまな憶測が飛び交った。ソフトウェアに焦点を置いているオラクルはパートナーや顧客に対する強引な戦術で知られており、Java コミュニティのメンバーの多くは同社が Java をより閉鎖的あるいは同社独自の方法で管理し、ライセンス料を引き上げるか、あるいは Java をオープンソースでリリースするという以前のサンの方向性を転換する可能性を懸念していた。欧州連合において最後に残っていた MySQL に関する規制上のハードルが解決した後、2010 年 1 月にオラクルはサンの買収の完了を発表した。それ以降、ほぼ 2 年半の期間における Java のリーダー企業としてのオラクルの影響を評価することが可能である。

Java の価値

オラクルは初期段階から Java の主要な支持者であり、IBM などと同様に、ミドルウェア製品のポートフォリオによって Java を中心として大規模のミドルウェア事業を構築していた。オラクルは Java の取得に先立つ 5 年間において大規模なソフトウェア買収攻勢に乗り出しており、これは 2008 年 4 月の BEA システムズ買収で最高潮に達していた。Java の買収に乗り出した時点では、オラクルはすでに IBM と並んで Java のミドルウェアの 2 大プレイヤーの 1 つであった。同様に重要なことであるが、オラクルは Oracle Application Development Framework（ADF）に基づいて同社のアプリケーション製品ラインの大規模なアーキテクチャ転換に投資を行っていた。これ自体、買収した複数のテクノロジー（たとえば、PeopleSoft や Siebel）を、Java を中心とした新しい包括的な製品スタックに融合させるものであった。この流れで見ると、サンの買収はオラクルがその将来を賭けたテクノロジーの所有を望んだ結果であると考えるべきである。

また、オラクルはサンのソフトウェアおよびハードウェアを活用した統合的なアプライアンス戦略の構築にも成功している。サンの買収により、オラクルはすでにリリースされていた Exadata に、Exalogic と呼ばれる Java ミドルウェアアプライアンスを追加した。オラクルの Exa シリーズは高い評価を受けており、その売上成長の大半は従来型の汎用 UNIX サーバーの売上に影響を及ぼすものとはいえ大幅な成長を達成している。全体的に見てオラクルはサンの統合に成功しており、ソフトウェアの統合を通じたハードウェア事業の価値の強化によって収益性と利益への大幅な打撃を回避し、サンの顧客に対するクロスセルによってソフトウェア事業の大幅な成長を達成している。多くの面において Java の取得はオラクルの戦略的に優れた施策であり、同社の市場および自社の戦略的な運命に対する支配

 ©2012 IDC

力を強化するものであった。オラクルにとってさらに重要な課題は、Java の価値の大きい部分を占める Java のエコシステムの全般的な健全性である。

Java コミュニティの不安

オラクルは長年に渡って攻撃的なイメージを培ってきた。このイメージは同社の超競争的なマーケティングおよび営業戦術に基づくものであり、また象徴的なリーダーであるラリー エリソン氏の発言に刺激されたものである部分も大きい。オラクルは以前からデータベーステクノロジーの革新者／リーダーとして認められており、休むことなく同社の主力製品に新機能を加えている。しかし、それと同時に同社は常に高価格のデータベースオプション（たとえば、Oracle Real Application Clusters、RAC）とライセンス保守料金によってそれらを積極的に収益化してきた。大多数の見方、特にオープンソースソフトウェアのコミュニティにおける見方では、オラクルは Java を運営するべき会社の対極的な存在であった。結果として、オラクルによるサンの買収は Java コミュニティに不安を引き起こした。オラクルが Java の実装をオープンソースに留めるのか、ライセンス料金の引き上げによって Java の収益化を積極的に進めるのか、あるいは Java のガバナンスをどのように行うのか、そしてその際にコミュニティの意見を十分に取り入れるか、などといった点が懸念の中心であった。したがって、過去 2 年半におけるこれらの分野でのオラクルの行動を評価することが重要である。

オープンソースコミュニティの不安の大部分は、おそらくは買収以前にサンが Java を効果的に収益化できなかったということに起因している。サンはハードウェアに焦点を絞った会社であり、その CEO がかつてよく語っていたようにソフトウェアは「ハードウェアの機能」と見なされていた。サンのハードウェアに対する集中によって、多くのソフトウェア企業が、サンには Java を中心とした伝統的なソフトウェア事業を構築する意思も能力もないと判断したことが、多くのソフトウェア企業が初期の Java の採用を促進する要因となった可能性が高い。実際、サンのハードウェアへの集中と大規模なソフトウェア事業の構築と運営における経験の欠如が、Java の運営とそのエコシステムの舵取りにおける極めて優柔不断なアプローチの原因となった。サンは多くの場合においてオープンソースの破壊的な性質と、ソフトウェアライセンスによって Java の効果的な収益化に必要な投資の間にはさまれて、そのいずれの面においても断固として、あるいは効果的に行動することはできなかった。これに加えて、市場はサンが製造、販売していた大型高性能サーバーから x86 サーバーに移行しつつあり、サンのハードウェア事業は緩慢な衰退の過程にあった。サンの管理下において、Java は進化と投資の停滞という問題を抱えていた。

JCP 設立の理由

広く利用されるテクノロジープラットフォームが、特定のベンダーの独断に従って進化することを回避するという課題は、Java にとって新しい問題ではない。Java はサンが独自の製品として開発、投入したものであるが、その広範な採用によって Java コミュニティは特定の営利企業への過度の依存に対して不安を抱くようになった。そのため、Java がコミュニティからのインプットと指針によって進化することを保証するために、Java Community Process（JCP）として知られるガバナンスプロセスがサンによって確立された。事実、JCP は Java を進化させるための一連の正式なプロセスから構成されており、その具体的な取り決めや範囲、複雑性の点で業界において他に類のないものである。このプロセスによって、相互に熾烈な競争相手である多くのベンダーのニーズを擦り合わせる形での Java の進化が可能になった。実際、ガバナンスプロセスとしての JCP は広大な Java エコシステムの構築に寄与しており、業界内での Java の位置付けにとって中核的な役割を果たし、オラクルや IBM、SAP、レッドハットなどの競合が単一の組織の下

で共通の利益のために協力することを可能としている。しかし、JCP に課題がないわけではない。

オラクルとオープンソース

オラクルがサンの買収以前においても、オープンソースのプロジェクトを推進し、またそれに参加していた事実は多くの場合において過小評価されている。オラクルは Eclipse Foundation の活動的なメンバーであり、自社の JDeveloper IDE をサポートしながらも、Eclipse IDE を積極的にサポートしている。もう 1 つの例は 2007 年のオラクルによる TopLink の Eclipse Foundation への寄贈である。これはその後、Java Persistence API 2.0 コンポーネントのリファレンス実装となっている。他にも、特に Linux、PHP、Apache、Eclipse、Berkeley DB、InnoDB などへの貢献や参加などの多くの例が存在する。サンの買収後、オラクルは Java を完全なオープンソースコードに移行するためのより積極的な措置を断行している。オラクルは 2011 年半ばに、リファレンス実装を OpenJDK プロジェクトのオープンソースベースにすると発表した。これは、オラクルの開発プロセスを簡素化すると同時に、コミュニティに対する透明性を高める一歩でもあった。この動きによって、オラクルは OpenJDK プロジェクトをより信頼性の高いものに転換し、それによってその後 IBM やアップルなどの協力者を勝ち取ることが可能となった。

買収後 2 年半が経過して、オープンソースコミュニティが恐れていたことはいずれも現実のものとはなっていないと言える。確かに、オラクルはサンの広範なオープンソースポートフォリオを縮小している。たとえば、OpenSolaris は早くから中止され、OpenOffice や Hudson などのプロジェクトを巡るオープンソースコミュニティとの衝突はコードの分裂を招き、最終的には他のオープンソース団体の管理下にコードを移管する事態になっている。しかし、全体として見れば、オラクルはより断固とした、また焦点を絞った Java のガバナンスを行っている。オラクルにおける Java の収益化の状況は買収以前とほぼ同様であり、（サンと同様に）ミドルウェアポートフォリオを通じて収益化が行われているが、ライセンス料は大幅には引き上げられておらず、それがオラクルのソフトウェア収益に占める比率はわずかであり、しかも低下傾向にある。実際、オラクルは OpenJDK プロジェクトにパートナーを迎え入れ、OpenJDK を Java SE のリファレンス実装に決定することによって、オープンソースへの注力を倍加させている。オラクルは Harmony の独自の Java 実装に対する検証キットのライセンス拒否など、いくつかの不評を買う決定を行ったかもしれないが、同社の決定は JCP がより早く、そしてより生産的な経路に沿って動くように促すことを目的としたものであった。

OpenJDK そして Harmony の終了

仕様リーダーとして、オラクルは新しいリリースのための Java のリファレンス実装を提供しなければならない。OpenJDK は、Java SE のコードを GNU 一般公衆利用許諾契約書（GPL）バージョン 2 の下でオープンソース化するためにサンによって 2006 年に開始された。OpenJDK は当初は高く評価されたが、Java のステークホルダーの一部は Harmony と呼ばれる Java の独立したクリーンルーム実装をサポートし始めていた。具体的には、IBM と Apache がライセンス料とサンによる強力な支配を伴うことなく Java をさらに広く提供することを意図し、より制約の緩い Apache Software License（ASL）の下での Java 互換のクローンの開発／リリースを目的として 2005 年に Harmony プロジェクトを開始した。

OpenJDK は歴史的にサンが単独で（レッドハットから一定の支援を受けて）維持していたプロジェクトであった。2010 年後半に、IBM とオラクルは OpenJDK コミュニティにおける協力を発表した。この動きは Java エコシステムにおける 2 大営利企業が OpenJDK に結集し、IBM を Apache Harmony プロジェクトのサポー

 ©2012 IDC

トから引き離すものであった。以前の IBM は Harmony を構成する主たる営利企業の 1 つであり、また資金提供者でもあった。そのため、この資金供給の停止によって Apache Foundation は 2011 年後半に Harmony を終了し、そのコードベースを Apache Attic に移行した。オラクルは 2011 年半ばに、リファレンス実装を OpenJDK プロジェクトのオープンソースベースにすると発表した。これは、オラクルの開発プロセスの簡素化と同時に、コミュニティに対する透明性を高める一歩でもあった。以前、商用の Sun JDK はリファレンス実装であり、これによってサンは他の実装に対する優位を確保していた。オラクルは、自社の商用実装も、他のベンダーと同様に従う必要のあるリファレンス実装として OpenJDK を選定することによって競争条件を公平なものとした。この動きによって、オラクルは OpenJDK プロジェクトをより信頼性の高いものに転換し、それによって IBM やアップルなどの協力者を勝ち取ることが可能となった。

ライセンスのない正式な Java のクローンを作成する決定は、オラクルがコミュニティの互換性のある単一の Java のバージョンへのコンバージェンスのために取った決定的な施策であった。これによって単一のガバナンス組織とブランドの下でコミュニティのエネルギーを単一のバージョンに集中させることが可能となり、細分化が回避されると共に、Java へのオラクルによる投資を確保する上で最も直接的な収益源であるライセンス料が確保されたことから、IDC ではこの動きが Java の進化にとってポジティブなステップであったとみている。この動きは、Java と部分的にあるいは完全に互換性を持つ類似したテクノロジーが今後出現しないということを保証するものではなく、また Java のエコシステムの細分化を阻止するものでもない。しかし、これは Java と呼ばれるすべてのテクノロジーが、概ね民主的な JCP の下で進化、管理されることを保証するものである。

競争と JCP の速度

JCP の重要な課題は、歴史的に意思決定プロセスに係るオーバーヘッドの大きさと、そのようなオーバーヘッドによるスピードの低下であった。Java は競争のない真空地帯で運営されているわけではない。業界ではアプリケーションプラットフォームを構築する数多くの試みがなされている。特にマイクロソフトのプラットフォームは、マイクロソフトの Windows OS と多様な Windows の派生 OS に集中したエコシステムの構築に成功している。マイクロソフトのプラットフォームの進化はそれほど民主的ではないが、生産性と市場投入時間の面でのメリットを得るため、展開の柔軟性や移植性をトレードオフとしてみる開発者を引き寄せている。また、もう 1 つのより広範で均質的ではないエコシステムが、より流動的な Web およびスクリプト言語アーキテクチャを中心に成長しつつある。このエコシステムは、リッチなブラウザーベースの JavaScript/HTML アプリケーションを中核とするものであり、背後にある Web サーバー上で PHP、Perl、Ruby、あるいは JavaScript など、数種類のスクリプト言語を動的に実行する。一部のケースでは、それらのサーバー言語自体も Java 上で実装されており、Java のエコシステムへの相乗効果や重複が生じているが、通常、これらの言語の開発者は自らを Java プラットフォーム開発者と考えていない。

コンピューティングの世界では、新しい言語やフレームワークが改良された新製品として絶え間なく発明／提案されているが、JCP 内の高度な民主主義は、Java がそのような世界での最先端の技術に遅れずに付いていくための必要経費となってきた。その反面、JCP は、Java が享受してきた幅広い企業における採用で必要とされるプラットフォームの一定の安定性を提供してきた。オラクル製品のマーケティングメッセージにおいて、Java の標準としての普及は重要な提供価値であるため、オラクルがこのプロセスを放棄する可能性は低い。したがって、オラクルは JCP を高速化するためにその過剰な官僚主義をスリム化しつつも、JCP の舵取りに責任を持たざるを得ない。

Java 7 の出荷

オラクルは JCP の課題を十分に認識した上で、買収以降の Java をより速く、より漸進的に進化させるための断固たる措置を講じた。サンは 2006 年末に Java Standard Edition 6 をリリースし、Java 5 プラットフォームに数多くの小さい改善や変更を加えた。売却交渉や買収承認の間にサンにおいて継続した引き延ばしのパターンは Java を急速に進化させるには不適切であった。そして、オラクルが 2010 年 1 月に所有権を引き継いだ後の混乱が収拾すると、サンが約束していた Java プラットフォームの改善は Java 7 と Java 8 の 2 つのリリースに分割された。実際には、オラクルは以前から JCP を通じてより漸進的なアプローチを主張しており、Java 7 の実装はすでに進行中であった。技術的な観点からは、Java 7 はネットワーク、セキュリティ、パフォーマンス、および国際化などの複数の分野における数多くの改善を含むものであるが、以下の 4 つが最も重要な新規投資の分野となっている。

- **動的言語のサポート（JSR 292）：**Ruby、Python、JavaScript など、近年において一定のマインドシェアを獲得している動的言語のサポートを改善するための InvokeDynamic バイトコード命令の追加（この新機能によって、これらの言語が高い効率とパフォーマンスで JVM を利用することが可能となり、これらの言語での開発者がより広範な多言語の Java エコシステムに集まる可能性がある）

- **File New IO（JSR 203）：**NIO.2 と呼ばれる、ファイルの非同期入出力処理のための新しいシステムと API セット（この新しい API によって、より広範なファイルメタデータへのアクセスや、エラー発生時におけるより多くの情報の提供が可能となり、現在注目を浴びているクラウドやビッグデータ環境内でのカスタムファイルシステムのサポートが提供される）

- **明示的なマルチコアのサポート（JSR 166）：**今日のマイクロプロセッサーにおけるコア集積度の上昇を活用する並列処理プログラムを開発者が明示的に作成できるようにするための新しい Fork／Join の API フレームワーク

- **Project Coin（JSR 334）：**Java 構文の更新、共通プログラミングタスクの簡素化を目的とした言語仕様の一連の強化（Java 7 での重要な追加構文には、コンパイラが［if-then-else 分の羅列と比較して］より効率的なコードを生成することを可能とする新しい switch 文、そしてプログラムの終了方法に関わらずクローズされるオブジェクトを宣言できる try-with-resources 文がある）

2010 年上半期の会社統合に関連した社内チームの再編成にもかかわらず Java 7 が 2011 年中盤に出荷されたという事実は、オラクルが目的をもって Java を管理していることを示している。したがって、Java 6 から Java 7 までの前例のない 4 年半の間隔は Java の背後にあるエネルギーの減速と見ることも可能であるが、オラクルは停滞を打破し、プロセスを前進させた点で評価されて然るべきである。

Java EE および軽量化へのシフト

Java Enterprise Edition は、大規模エンタープライズアプリケーションが伝統的に必要とする強力なセキュリティ、分散コンピューティング、データアクセス機能を提供するサービスフレームワークによって Java を拡張したプラットフォームである。Java EE は、Java SE ランタイム上に構築された、Java Professional Edition のアドオンフレームワークとして 1998 年に投入された。Java EE は企業の要件に対処するための CORBA などの重量級のアーキテクチャのエミュレートを目指し、

 ©2012 IDC

5 つの主要リリースによる改善が行われている。この過程において複雑性は増大したが、最後の 2 つのリリースではその一部が解消され始めている。

Java EE は、その一部の機能を必要とするアプリケーションを構築するために広く採用されているが、単一のアプリケーションにおいてその機能のすべてが要求されることはまれである。Java EE はリリース初期から、複雑さ、広大な範囲、メモリーとディスクのフットプリント、そしてスタートアップと展開のスピードの点で批判の対象となっていた。Java EE はやがて、軽量型アプローチで問題解決を試みようとする、SpringSource（現在はヴィエムウェアのアプリケーションプラットフォーム部門）の創立者である Rod Johnson からのオープンソースフレームワークとしてリリースされた Spring Framework による挑戦を受けた。また、Java EE は、Java EE 仕様の Web およびサーブレットのみに焦点を合わせた（たとえば、Apache Tomcat）、軽量で部分実装型の Java アプリケーションサーバーからの挑戦も受けた。実際、軽量コンテナフレームワークおよび Java EE の部分実装は、大企業における Java 採用の拡大に寄与し、Java EE の後期リリースにおける軽量アプローチと革新の要因となった（たとえば、Java EE 5 における IoC（制御の反転）や Java EE 6 における Web プロファイルを通じた限定的なモジュール化など）。

全般的に見ると、Java EE に対するこれらの挑戦は Java のエコシステム全体の強化に寄与しているが、Java EE アプリケーションサーバーの重量級機能（トランザクションサポートなど）の採用の拡大は近年になって減速している。この減速の一部はオープンソースのアプリケーションサーバーからの挑戦によるものであるとみられるが、エンドユーザーが接するアプリケーションのフロントエンドに対する関心の広がりや、より広範な Web テクノロジーの利用などを含む、大企業で開発されるアプリケーションの性質の変化も大きい要因になっている。

将来の展望

アプリケーション開発の現状は、コンピューティング環境の進化に応じて、重要な以下に示す 4 つの変化に起因するアーキテクチャ上の大転換の最中にある。

- **モバイル：**モバイル機器（スマートフォン、タブレット、PC、テレビ、セットトップボックスなど）の普及による、タッチ、ジェスチャー、音声インターフェースを伴う新しいインタラクションのパラダイム、および新しい配布方法と収益化の手法を伴う新しいスタイルのアプリケーションの登場

- **クラウド：**クラウドコンピューティングアーキテクチャとクラウドアプリケーションプラットフォームの登場による、バックエンドアプリケーションの配備における複雑さの軽減と、クラウド環境用に新規設計されたアプリケーションから構成されるスケーラブルな新規サービスの構築

- **ソーシャル：**ソーシャルネットワーキングプラットフォームや、エンドユーザが利用するソーシャル機能の優位性は、収集されたデータを通じ、新しいインターネットサービスや、それらを収益化するための新しい方法を作成（ソーシャル機能は、新規アプリケーションあるいは既存アプリケーションへの付加機能として企業に浸透しつつある）

- **ビッグデータ：**急増するデータと、その収集、保管、可視化、分析、取捨選択のニーズ、および分析に基づいた手動または自動化された意思決定をサポートする必要性（デバイスおよびデバイスセンサーの増加、そしてこれらのデバイスやそれをサポートするバックエンドシステムとのユーザーのやり取りの密度の上昇によってデータが生成されている）

Java が過去 10 年と同様に今後 10 年においても重要であり続けようとするのであれば、この変化のペースに対して Java を進化させる積極的な戦略が必要である。Java はこれらの領域の多くですでにプレイヤーであるという点に注意が必要である。Java のスキルは Android プラットフォームとそのエコシステムの進化で使用されており、Java は多くの PaaS（Platform as a Service）プレイヤーに提供されている。多くのソーシャルアプリケーションは Java で書かれており、重要なビッグデータテクノロジーの 1 つである Hadoop も Java で書かれている。しかし、前述のテクノロジーの進化のペースは容赦なく加速している。上述した変化に完全に対処できるように Java を進化させることは、営利企業による巨額の投資と Java コミュニティの膨大なエネルギーを必要とする難事である。オラクルは、この複雑かつ多様なアプリケーションの状況を考慮して、現在果たしている重要な役割を維持できる Java のビジョンを追求する必要がある。

SE 8：並列処理とコンバージェンス

オラクルの真価が試されるのは、現時点において 2013 年中盤と目されている Java 8 の実現である。概ね期限通りにリリースされたとしても、Java 8 の機能はほぼ 7 年間も実装が待たれていたものである。Java 8 は仕様定義の最終段階にあり、2013 年後半のリリースに向けたスケジュールが組まれている。Java 8 では多くの機能強化が行われる予定であるが、リリースの重点として最も期待されている分野は、モジュール化、ラムダ関数構文、そして JVM コンバージェンスである。

- **Project Lambda の関数型プログラミング：**Java 8 では、Project Lambda において、関数型プログラミングの構造が導入される。関数型プログラミングはクロージャとしても知られるラムダ関数で記述される計算の宣言的なスタイルを特徴としている。計算が状態遷移の命令によって明示的に表現されるのではなく、機能的に表現されることによって、基盤となるシステムはより柔軟に並列処理を利用することが可能になる。また、ラムダ言語の変更によって、filter／map／reduce として知られる、大規模なデータセットを処理するための操作を Java に加えることが可能になる。ムーアの法則によって、トランジスター集積度はかつてない水準に達し、開発者が従来の直列型プログラミングでそれを活用することが困難となっているため、並列処理サポートの改善が必要となっている。Java における関数型プログラミングの提供によって、開発者は新しいハードウェアアーキテクチャの処理能力を最大限に活用することが可能になる。

- **JVM のコンバージェンス：**サンの買収に先立って、オラクルは BEA を買収し JRockit JVM を継承している。JRockit JVM はオラクルの戦略的ミドルウェアのための JVM となっている。また、サンの買収により、オラクルはより広く採用されている JVM である HotSpot も継承している。Java 8 では、JRockit JVM と HotSpot の両方から最高の機能を統合した新しい JVM が約束されている。たとえば、JRockit の Mission Control や Flight Recorder などの保守性機能は Java 8 に組み込まれると見られる。オラクルではまた、少数の実装に対して広範な機能を提供することに集中できるように、サンの複数の実装の一部を統合することを示唆している。Java のパフォーマンス、信頼性、セキュリティは JVM の改良に依存しているため、JVM のコンバージェンスは困難な課題である。

SE 9：モジュール化の延期

- 以前において SE 8 の目標とされていた Java の重要な改善の 1 つが新しいモジュールシステムである。オラクルでは最近になってこのプロジェクトの作

 ©2012 IDC

業が SE 8 の変更に含めるほど十分には進捗していないと判断し、この機能は SE 9 で提供されることになった。

- モジュール化とは、アプリケーションにおいて Java ランタイムの一部をファクタリングできる機能を指している。モジュール化は、ソフトウェアアーキテクトが熱望するソフトウェア開発における理想である。JDK などの大規模なコードは過去 15 年間に成長しており、極めて複雑で相互に接続されている。Java のシステムライブラリーを相互に絡み合っていないモジュールグループに整理する作業は、2008 年に Project Jigsaw で最初に計画された野心的な作業である。Java のアーキテクチャをモジュール化すれば、ソフトウェアの複雑さを軽減し、Java のプロファイルを分けることによって、多様なフォームファクターに優れた互換性を提供できる。

- Project Jigsaw を SE 9 にシフトさせたのは正しい決断であり、オラクルが機能実装のスケジュールに優先順位を設定したのは適切であった。異なる開発者がプラットフォームに異なる機能を求めているが、安定したリリースの提供が必要であるという点ですべての開発者が合意している。Java エコシステムの大部分は極めて急速な変化を吸収することはできないが、言語は技術革新を促進するのに十分な速度で進化する必要がある。開発プロセスの成熟度は、そのスケジュールの予測可能性で測定される。モジュール化は困難な問題であり、メリットは間接的であるが、Java エコシステムを拡大し Java をさらに多様な環境に広げるという面で潜在的なメリットは膨大である。しかし、Java は競争のない真空地帯で運営されているわけではなく、さらに軽量なテクノロジーが代替技術として絶え間なく Java に挑戦している。クラウドサービス、ソーシャルインタラクション、モバイルアプリケーションストアの時代には、より速いペースでの進化が必要であり、すべての開発において 2 年のサイクルをより段階的で高速なアプローチに転換しなければならない。

EE 7：マルチテナンシー化の機会は不透明

Java Enterprise Edition は歴史的に 3 年サイクルで出荷されている。Java EE 6 は買収完了の 1 か月前の 2009 年 12 月に出荷されている。現在、Java EE 7 は後期フェーズにあり、2012 年末までにリリースされる予定である。EE 7 には数多くの機能追加と変更が含まれているが、重点の 1 つがクラウドコンピューティングのサポートである。サーバーコンピューティングはクラウドコンピューティングに進化している。これは仮想化で細分化されたリソースが、高度に自動化されたプロビジョニングと管理を通じて開発者に提供されることに依存している。クラウドコンピューティングは、多くのコンシューマーアプリケーションに機動的なバックエンドを提供することによって、オンライン、ソーシャル、モバイル革命を支えるテクノロジーとなっている。クラウドアプリケーションプラットフォームの採用は現在急激に増加しているが、その利用は主に新世代の Web、ソーシャル、モバイルアプリケーションのバックエンドの領域に限られている。企業はまだクラウドプラットフォームの大規模な利用に動いていない。Java EE 7 では、今後 5 年間に予想される、企業におけるクラウドアプリケーションプラットフォーム採用の波に伴う今後の変化のサポートが計画されている。

Java EE 7 プラットフォームの中核的機能の 1 つとして注目されているのは、マルチテナンシーをサポートするためのワークロード分離の概念である。Java アプリケーションの分離は JVM の仮想インスタンスとロードされた Java EE クラスによって達成することが可能であるが、これは極めて細分化されたリソースを必要とする新しいクラウドアプリケーションにとって粒度が非常に粗く、リソース集約的なソリューションであると広く考えられている。大半の PaaS で見られる重要な点は、フリーミアム（基本的なサービスは無料で提供し、より高度な機能や付

加価値の高いサービスなどについては有償で提供）のビジネスモデルにおいて多数の開発者をサポートすることであり、複数のアプリケーションが単一の仮想マシンで安全に実行できるよう設計されていない限り、コストが増大する可能性があるということである。Java EE でのマルチテナンシーは、大規模な Java クラウドプロバイダーの運用に必要とされる粒度をサポートするために、このシナリオの実現を意図したものである。

マルチテナンシー環境での信頼性と安定性のサポートは困難な課題であり、この機能の実現は重要である。コミュニティでは、EE 7 におけるクラウド機能への過度の注力によって、プラットフォームの他の側面における進歩が犠牲になるという懸念が一部で見られる。IDC では大企業における新しい EE のリリース採用は極めて緩慢であり、多くの企業では EE 6 の利点もまだ活用されていないとみている。Java EE は引き続き急速なペースで進化する必要があるが、EE 7 の重点としてクラウドコンピューティング機能の実装に時間をかけるのは適切である。

IDE の投資中断はない

サンの買収により、オラクルは比較的定評の高い NetBeans IDE を継承した。IDC では、Java 開発者間の人気の面で NetBeans を上回るのは Eclipse だけであるということを多くの場面で確認している。NetBeans はオープンソースで汎用的な Java IDE として、オラクルの JDeveloper IDE よりも高い人気を有しているが、JDeveloper IDE は引き続き Oracle Fusion Applications および Oracle Fusion Middleware のための主たる IDE となっている。オラクルは賢明にも NetBeans への投資を継続し、さらにその人気を高め、採用ベースを拡大している。NetBeans は、JavaScript や PHP などの一般的なスクリプト言語向けだけではなく、エンタープライズモビリティ機能の開発においても支持されている。

また、オラクルでは Eclipse もサポートしており、大規模な Eclipse 開発者コミュニティがオラクルのスタックで生産的に作業できるように、Oracle Enterprise Pack for Eclipse を提供している。また、オラクルでは Oracle Database との作業を行うマイクロソフトのエコシステムの開発者に対するサポートを行っていることも評価に値する。特に、Oracle Developer Tools for Visual Studio は、マイクロソフトの Visual Studio のアドオンであり、開発者が Visual Studio において MS SQL Server データベースを扱うのと極めて類似した方法で Oracle Database を扱うことを可能としている。

IDE はソフトウェアベンダーの研究開発に基づく収益化の中心ではなくなってから久しく、プラットフォームの実行環境への投資に変化している。複数の IDE への資金投入はオラクルにとって長期的には高価につく可能性もあるが、IDE の習得にスキルを投入した開発者からの極めて大きい信用の獲得につながっており、この投資は適切なものとなっている。

Android の影響

Linux から派生したオペレーティングシステムである Android は、モバイルおよび組み込みアプリケーション向けの Java を脅かし始めている。グーグルは新しいスマートフォンプラットフォームへの牽引力を最大化するために、Java 言語でのプログラムが可能なプラットフォームを作成することにより、Java 開発者のエコシステムを利用した。グーグルは、オラクルから Java をライセンスしておらず、Android プラットフォームの記述でも「Java」という言葉を使用していない。その代わり、グーグルでは、Dalvik という名称の新しい仮想マシンのクリーンルーム実装を使用していると主張している。Dalvik では、アプリケーション開発者が作成した Java 言語のコードから生成される異なった仕様のバイトコードが実行され

 ©2012 IDC

る。膨大な数の開発者が Java に精通しているという事実が、Android において Java が選択された最大の要因である可能性が高い。

グーグルでは Android を定義する過程において、Android アプリケーションが Java と非互換になる形で、多くの API を異なった実装としている。多くの関係者は、Android によって Java の開発者が成長過程にあるスマートフォンアプリケーションに関わることが可能となったことから、Android が Java エコシステムにとってポジティブな展開であると考えた。しかし、Android は Java の互換性に関するメッセージを否定しており、時間の経過と共に他の利用分野にも拡大する可能性の高い別のグループを形成している。Android はすでにタブレットに拡大しており、将来は Android の完全な PC が登場し、Android アプリケーションのエコシステムを土台として利用して Windows PC と競合する可能性もある。Android は Java のエコシステムにとってプラスの資産であるとみている向きも多いが、Android は Java 分裂のリスクとなっている。Android は 2 つの点で Java と大きく異なっている。第 1 に、Android はグーグルによってより独断的に支配されており、コードの初期のマイルストーンはクローズドソースに留められているという点である。もう 1 つは、Android がハードウェアメーカーには無料のオープンソースとして提供されている点であり、これが急速な採用拡大の重要な要因となっている。Java が直面している脅威は、Android ベースの並行的に競合するフレームワークが、最終的に ARM ベースのサーバー上などのモバイルフォームファクター以外で Java と競合する事態である。Java 対 Android の状況は、Java のエコシステムを巡る競争圧力の結果として、グーグルとオラクルが今後 2 年間に取る行動にかかっている。

IDC の提言

Java は膨大な経済市場を支えている。IDC ではその規模を明示的に推定したことはないが、それはハードウェア、ソフトウェア、サービスで年間 1,000 億ドルを上回るとみている。そのエコシステムの価値は、存在する製品やサービスの豊富さに反映されており、いったん質量共に臨界点に達すると、そのテクノロジーのユーザーが切望する長期的な安定性と安全性を提供することができる。Java の場合、開発者は正に Java エコシステムの規模ゆえに、Java とそのエコシステムを中心としたスキルの習得と構築に安心して投資することが可能である。急速に変化するテクノロジー概況を前に、ソフトウェアのエコシステムの長期的な健全性を維持することは困難な課題であり、それはエコシステムのリーダーが担わなければならない。Java の場合には、そのリーダーは今やオラクルである。

参考情報

関連調査

- *Worldwide Development Languages, Environments, and Tools 2011 Vendor Shares*（IDC #235129、2012 年 6 月発行）
- *Worldwide Enterprise Portals 2011 Vendor Shares*（IDC #235128、2012 年 5 月発行）
- *Going Mobile: The Coming Convergence of Front-End Application Platforms and Ecosystems*（IDC #234000、2012 年 3 月発行）
- *Application Development Trends: Developer Ecosystems in Flux*（IDC #DR2012_BB1_AH、Directions 12 Presentation、2012 年 3 月発行）

- *Windows 8 on ARM: Important New Disclosures*（IDC #lcUS23318012、2012 年 2 月発行）
- *IDC Predictions 2012: Application Development & Deployment*（IDC #WC20120117、2012 年 1 月発行）
- *Windows 8 Application Development Development & Deploymention*（IDC #230687、2011 年 9 月発行）
- *Worldwide Development Languages, Environments, and Tools 2010 Vendor Shares*（IDC #229273、2011 年 7 月発行）
- *Getting Old Apps to Windows on ARM: Not So Easy*（IDC #lcUS22873911、2011 年 6 月発行）
- *Worldwide Application Development Software 2011and Tools 2010 Vendor Shares Ecosystems る べ*（IDC #229150、2011 年 6 月発行）
- *Worldwide Web Design and Development Tools 2010 Vendor Shares*（IDC #228971、2011 年 6 月発行）
- *Worldwide Application Development and Deployment 2011 Top 10 Predictions*（IDC #227110、2011 年 3 月発行）
- *Microsoft Gets Its Mobile Mojo Back with Windows Phone 7*（IDC #lcUS22642610、2010 年 12 月発行）

Synopsis

本調査レポートでは、オラクルによるサン・マイクロシステムズの買収から 2 年半後における Java の状況の評価を行っている。

IDC アプリケーション開発ソフトウェアリサーチ担当のプログラムディレクターである Al Hilwa は、「すべてのソフトウェアプラットフォームにおいて、その成功の基準は、協調的かつ競争的に付加価値を構築し、開発者、ベンダー、テクノロジーからなるエコシステムを育成できる能力にある。Java は現代のソフトウェアにおいて存在する最大級のエコシステムの構築に成功している」と述べている。

Copyright Notice

This IDC research document was published as part of an IDC continuous intelligence service, providing written research, analyst interactions, telebriefings, and conferences. Visit www.idc.com to learn more about IDC subscription and consulting services. To view a list of IDC offices worldwide, visit www.idc.com/offices. Please contact the IDC Hotline at 800.343.4952, ext. 7988 (or +1.508.988.7988) or sales@idc.com for information on applying the price of this document toward the purchase of an IDC service or for information on additional copies or Web rights.

Copyright 2012 IDC. Reproduction is forbidden unless authorized. All rights reserved.