様　式　Ｃ－１９、Ｆ－１９、Ｚ－１９　（共通）

科学研究費助成事業　　研究成果報告書

平成 ２７ 年　 ６ 月　 ９ 日現在

機関番号：１４３０１
研究種目：若手研究(B)
研究期間：2012～2014
課題番号：２４７０００２８

研究課題名（和文）アンビエント計算に基づく実用的かつ信頼性の高い分散プログラム開発環境

研究課題名（英文）A Develop Environment of Practical and Reliable Distributed Programs Based on the Ambient Calculus

研究代表者
馬谷　誠二（Umatani, Seiji）

京都大学・情報学研究科・助教

研究者番号：４０３７８８３１

交付決定額（研究期間全体）：（直接経費）　　3,400,000円

研究成果の概要（和文）：本研究では，研究代表者がこれまでに開発を行っていたアンビエント計算に基づく分散プログラミング言語から得られた柔軟な分散協調機能の設計に関する指針およびそれらの機能の効率良い実現手法を活用し，汎用プログラミング言語を用いて信頼性の高い分散プログラムを開発するためのフレームワークの開発を行った．このフレームワークは，実用的かつ柔軟性の高いアンビエント計算機能を提供するのに加え，専用分散言語で書く際には問題となりにくいユーザプログラムの安全性を保証するための解析・検証ツールを備えている．

研究成果の概要（英文）：In this research, we developed a framework for developing reliable distributed programs in general purpose programming languages, by following the guideline about the design of several flexible cooperation facilities in distributed environments and by exploiting their efficient implementation techniques, both of which are derived from our previous distributed proggramming language based on Ambient calculus. This framework provides practical and flexible ambient-based features. Furthermore, it also supports a tool for analyzing and verifying the security of user programs, which would be less problematic if we wrote such programs in domain-specific distributed languages.

研究分野：プログラミング言語

キーワード：分散プロセス計算　プログラミング言語　コード移動　プログラム解析

１．研究開始当初の背景

アンビエント計算は，分散計算のためのプログラミングモデルとして重要な実行主体（プロセス），場所，移送単位の概念を,階層的なアンビエントとして表現する強力な計算体系である．すべてを統一的に表現するため，シンプルでありながら柔軟な表現力も備えており，これまでファイアウォールや大規模分散システムの安全性などの研究に数多く用いられている．
一方，アンビエント計算は直観的であり柔軟な表現力を備えているにも関わらず，現実のプログラミング言語にその機能が直接的に取り入れられることは，ほとんどなかった．ただし，現実の分散環境上でアンビエント計算を動作させるための要素技術に関する先行研究はいくつか存在した．
そのような状況の中，近年，研究代表者はアンビエント計算の機能を備えた高水準プログラミング言語の開発を行ってきた．プログラマは，高水準なアンビエント計算の機能を用いてプログラムを書けるため，複雑な同期通信や協調動作を必要とする大規模な分散プログラムをより簡潔かつ直観的に書くことができる．学術的には，高水準言語の開発を通じて：

- 抽象的なモデルではなく現実のプログラムを書くために,アンビエントの概念をどのように用いるべきか，あるいは，もとのアンビエント計算にどのような拡張・修正が必要となるか
- 現在主流となっているヘテロかつアドホックな大規模ネットワーク環境においても,確実にかつ効率良くアンビエント計算を動作させるための実装手法
- 従来のフラットなネットワークと通信チャネルによる分散プログラミングと比較し,生産性・保守性の向上を図れることを例証

などの知見・技術を得られた．

２．研究の目的

(1)　概要

分散計算モデルに基づく専用の高水準言語を使って，世の中すべての分散プログラムを書くようにするというのは，現実的には難しい．少なくとも分散処理に依存しないコンポーネント内の閉じた処理については，その処理に適した既存の汎用言語で書ける方が多くのプログラマにとっては望ましいと言える．そこで本研究では，汎用言語用のアンビエント計算ライブラリを開発し，より多くのプログラマが一般には敷居が高いと思われている分散プログラムを容易に書けるようにする．ライブラリの設計を上手く行い，前述の先行研究の成果である高水準言語と同程度の柔軟性・生産性・保守性を備えることが目標の一つである．
汎用言語用ライブラリにおいては，その振舞いの正しさの保証が大きな問題となる．分散プログラムの諸性質を高水準なモデルで簡潔に表現できたとしても，汎用言語で書く部分のコードとの干渉により，その性質が損われてしまったのでは，正しいプログラムを開発することはできない．そこで，本研究では，単に汎用言語のコードから呼び出せるライブラリを作成するだけではなく，ライブラリを使用するコードがプログラマの意図（アンビエント計算で表現する設計仕様）から外れる動作をしないことを保証するための手段（フレームワークによる制限，性質の解析・検証ツール）も同時に提供することにする．具体的には，以下の 3 つを研究目的として挙げる．

① 現実の大規模分散プログラムの設計・コーディング・解析・検証の全てをアンビエント計算モデルに基づいて統一的に行うための基盤を構築する．
② 分散プログラムのコンポーネント（アンビエント）分割およびコンポーネント間の安全な協調動作を実現するための要素技術の確立．
③ 汎用言語で書くことによる利点を損ねないために，オブジェクト指向機能・関数型機能とアンビエント計算の機能の料率（あるいは統合）を可能にするフレームワークの開発を行う．本研究中では得に，静的型付オブジェクト指向言語 Java（あるいはより洗練された型システムを持つ Scala）と動的型付関数型言語 Scheme の 2 つを具体的なターゲットとする．

(2)　学術的な特色および研究の意義

本研究は，学術的に以下のような特色を持つ．

① 高水準仕様，現実のコード，性質記述が全て同じ形式モデルに基づいて記述されるため，モデルの不一致による謝りの混入を防ぐことができ，正しく安全なプログラムの開発を促進する．
② 階層化された形式モデルはプログラムの柔軟な分割を可能にする．移送機能と合わせ，段階的なプログラムの進化や大幅な構造修正への柔軟な対応を可能とする分散システムの強力な開発基盤となる．
③ 分散システムは階層化されたアンビエントの集合として構成されるため，各アンビエントの解析・検証結果から合成的にシステム全体の性質を導出できる．

本研究は，クラウドコンピューティング，P2P コンピューティング，モバイルエージェントシステムなどに代表される多種多様な分散システムを構築するための新な開発基盤を提供する．明確な分散モデルを持たない汎用言語のみでシステムを構築する場合と異なり，形式的なモデルに基づく安全なコードを動作させることの恩恵は大きい．さらに，同時提供される解析・検証ツールは，開発プロセスにおけるバグの大幅な削減を可能にす

るとともに，大規模分散システム開発全体における生産性の大幅な向上をもたらすと期待できる．

３．研究の方法

本研究では，以下のシスム・ツール・手法を順に開発することで，目標とする開発環境の構築を目指した．

(1) 高水準言語向けライブラリの開発
(2) セキュリティ仕様記述のための言語の設計
(3) 低水準コードの解析手法の確立
(4) アンビエント計算プログラムのセキュアな実行環境の開発

４．研究成果

(1) 高水準言語向けライブラリの仕様策定およびプロトタイプ実装
分散プログラムの柔軟な開発を目的とした，Scheme 言語用ライブラリの開発を行った．開発した実行環境は，対話環境を備え，プログラムを実行させながら，アドホックなコードの追加，移動，マージ等のアンビエントの構造的な変更を柔軟に行うことが可能である．開発したライブラリは，移動を実現するための実装方式の提案，アンビエント階層可視化ツールの開発，関数型言語の持つ高階関数，スコープルールとの兼ね合いを考慮した仕様の設計等，高水準言語上でアンビエント計算モデルを用いてプログラムを記述するのに有用な機構を数多く備えている．

(2) 高水準言語向けライブラリの分散環境上における安全な実装技術の開発
アンビエントにおいて個々の移動動作を表すケーパビリティは，秘匿性と不可分性の 2 つの性質を備えている．秘匿性とはケーパビリティから動作対象の名前を抽出できないこと，不可分性とは，合成されたケーパビリティを個々の要素に分解できないことを意味する．これらの性質により，アンビエント間のやり取りに厳密な制限を課すことが出来，システムの安全性を保証しやすくなる．アンビエント計算の実装において，これら 2 つの性質を備えたケ ーパビリティを実現するための手法について提案する．提案手法は，公開鍵暗号方式を用い，適切な場所でのみケーパビリティを復号可能とすることで性質を保証する．提案手法の形式的定義を示し，また，(1)で開発したシステム上に実装した．

(3) 高水準アンビエント記述(アプリケーションプログラム)に対するセキュリティポリシー記述仕様の確立
高水準記述が可能な Java フレームワーク上で記述された分散アプリケーションに対し，移動先で実行され得るケーパビリティの指定(コードからの抽出が容易でない部分の事前埋め込み)と，受け入れ側のセキュリティポリシーの指定方法の仕様記述の設計を行った．

(4) 低水準コードの解析手法
(3)の高水準フレームワーク上で記述あれた Java プログラムは，標準的な Java コンパイラによって低水準なバイトコードへと変換されるが，アプリケーション実行中の安全性の確保の実現には，そのような低水準コードの実行時解析・検証が必要となる．本課題で開発しているアンビエント計算機能を用いた Java コードの特性を考慮した簡潔な解析手法，および解析を実装するためのバイトコード操作ツールの開発を行った．このツールを使えば，Java 仮想機械上で動作するアンビエントプログラムの安全性を，プログラムの実行時ではなく，クラスロード時に検証するための手法として Java バイトコードに対する低水準の解析・操作を柔軟かつ簡潔に記述可能である．これにより，アンビエントプログラムの安全性に関する様々な記述を，このツールを用いる Java プログラムへとコンパイル・実行することで，開発基盤のセキュリティに関するコンポーネントの実装の改善を見込むことが可能となる．

(5) 高水準言語に組込まれたアンビエント機能のセキュアな実行のためのフレームワークの開発
アンビエント計算において個々の移動動作を表すケーパビリティは，処理系のセキュリティを確保するための重要な役割を持っており，その正しくかつ効率良い実行方式の実装は重要な課題である．そのような実行方式の一つとして，動的な機能をそなえた汎用プログラミング言語へのコンパイルによる機能の実装を行った．これまでの研究成果であった解釈実行による方法と比べ，実行速度の改善が見られ，かつ，形式的に記述された変換方式を提供することで，セキュアな実装が正しく行われていることの確認がより容易になっている．

５．主な発表論文等
（研究代表者、研究分担者及び連携研究者には下線）

〔雑誌論文〕（計５件）

① Seiji Umatani, Tomoharu Ugawwa, Masahiro Yasugi, Design and Implementation of a Java Bytecode Manipulation Library for Clojure, Journal of Information Processing, 査読有，印刷中，2015
② Seiji Umatani, Practical

Implementation Techniques of Ambient Calculus in Conventional Dynamic Languages, Proceedings of the 29th Annual ACM Symposium on Applied Computing (SAC2014), 査読有, 1345-1351, 2014
DOI: 10.1145/2554850.2554995

③ 田附 正充, 八杉 昌宏, 平石 拓, 馬谷 誠二, L-Closure の呼び出しコストの削減, 情報処理学会論文誌 プログラミング, 査読有, Vol.6, No.2, 13-32, 2013

④ 林 奉行, 馬谷 誠二, 八杉 昌宏, 湯淺 太一, Safe アンビエントに基づく分散アプリケーション開発用 Lisp 環境, コンピュータソフトウェア, 査読有, Vol.30, No.1, 231-256, 2013
DOI: 10.11309/jssst.30.1_231

⑤ Haruna Nishiwaki, Tomoharu Ugawa, Seiji Umatani, Masahiro Yasugi, Taiichi Yuasa, SEAN: Support Tool for Detecting Rule Violations in JNI Coding, IPSJ Transaction on Programming, 査読有, Vol.5, No.3, 23-28
DOI: 10.2197/ipsjtrans.5.139

〔学会発表〕(計９件)

① 馬谷 誠二, 鵜川 始陽, 八杉 昌宏, Clojure 用 JVM バイトコード操作ライブラリの設計と実装, 情報処理学会第 102 回プログラミング研究会, 2015 年 1 月 13 日, 宮崎大学木花キャンパス(宮崎県)

② Seiji Umatani, Practical Implementation Techniques of Ambient Calculus in Conventional Dynamic Languages, 29th Symposium on Applied Computing (SAC 2014), 2014 年 3 月 24 日, Gyeongju, Korea

③ 馬谷 誠二, JVM バイトコードへの低水準操作を簡潔に記述可能なマクロシステム, 第 55 回プログラミング・シンポジウム, 2014 年 1 月 10 日, ラフォーレ伊藤(静岡県)

④ 馬谷 誠二, 行儀の良い分散プログラミングのための拡張 Lisp 言語, 数理システム Lisp セミナー, 2013 年 11 月 21 日, NTT データ数理システム(東京都)

⑤ 田附 正充, 八杉 昌宏, 平石 拓, 馬谷 誠二, L-Closure の呼び出しコストの削減, 情報処理学会第 92 回プログラミング研究会, 2013 年 1 月 15 日, AiAi ひろば(鹿児島県)

⑥ Haruna NIshiwaki, Tomoharu Ugawa, Seiji Umatani, Masahiro Yasugi, Taiichi Yuasa, Detecting Bugs in Android Using a Static Escape Analyzer SEAN for Native Code, 10th Asian Symposium on Programming Languages and Systems (APLAS 2012), 2012 年 12 月 11 日, Kyoto International Community House (Kyoto)

⑦ 馬谷 誠二, 八杉 昌宏, Safe アンビエントの移動動作のセキュアな実装手法, 日本ソフトウェア科学会第 29 回大会, 2012 年 8 月 22 日, 法政大学小金井キャンパス(東京都)

⑧ 松井 健, 平石 拓, 八杉 昌宏, 馬谷 誠二, ワークスティーリングフレームワークにおける集団通信機能, 2012 年並列/分散/協調処理に関する『鳥取』サマー・ワークショップ(SWoPP 鳥取 2012), 2012 年 7 月 31 日, 北九州国際会議場(福岡県)

⑨ 松井 健, 平石 拓, 八杉 昌宏, 馬谷 誠二, 高速版 Barnes-Hut 多体シミュレーションの並列実装, 先進的計算基盤システムシンポジウム(SACSIS2012), 2012 年 5 月 16 日, 神戸国際会議場(兵庫県)

６．研究組織

(1)研究代表者

馬谷　誠二（UMATANI, Seiiji）

京都大学・大学院情報学研究科・助教

研究者番号：４０３７８８３１