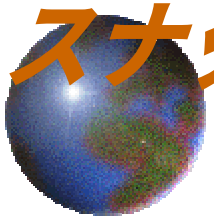
UMLスクラップブックと スナップショットプログラミング環境の実現

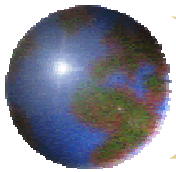


佐藤治

Richard Potter

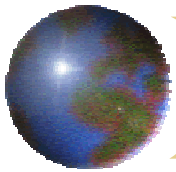
山本光晴

萩谷昌己

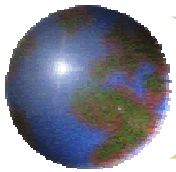


Overview

- SBUML開発の背景と動機
- SBUML実装の手法
- SBUMLの実装
- SBUML Demo.
- SBUMLインターフェース
- スナップショットプログラミングとSBUMLモデル検査システム
- 今後の課題･まとめ

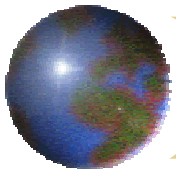


Introduction



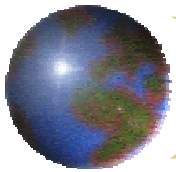
Introduction – SBUML As Our Work

- User-mode Linux(UML)**に対する拡張**
  - Checkpointing**関連の機能**
    (サスペンド、スナップショット作成)
  - **関連するその他の機能**
    (状態抽出、実行制御)
  - **ユーザに**programming interfaces**を提供**



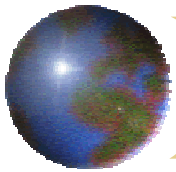
Introduction – Two Motivations(1)

- **仮想計算機システムの実用性を高める新たな実装として**
  - **オープンなソフトウェア上での**Checkpointing**機能の実装 (コミュニティへの貢献、**etc.)
  - Checkpointing**機能付き仮想計算機を**
    1つのモジュールとしてプログラマに提供することで、
    Checkpointing**機能を外部から利用した**
    アプリケーションを記述可能にする
    (**スナップショットプログラミング**)



Introduction – Two Motivations(2)

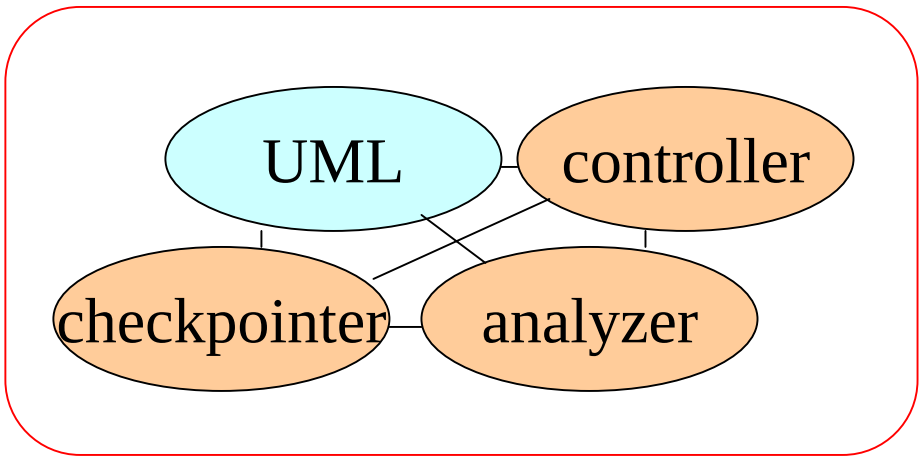
- 「ソフトウェアモデル検査システム」実装の1つのアプローチとして
  - 「モデル検査」とは、プログラムのバグを自動的に検出する技術の一つ
  - 「ソフトウェアモデル検査」では、実行可能形式のプログラムに対する直接的検証を行う
  - 検証対象の実行状態を自在に把握、操作できる環境の構築



Introduction – Features of SBUML

- **任意の時点でのサスペンドが可能**
- **任意の実行状態のスナップショット化及びスナップショットからのUML再実行** (Checkpointing)
- **モデル検査に必要な各種機能の提供**
  - **スナップショットからの状態抽出**
  - **UMLスケジューラ制御**
- **SBUMLインターフェースの提供**
  - **外部スクリプトからのSBUMLの利用**

Introduction – SBUML Architecture

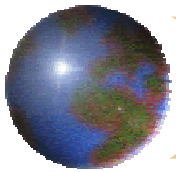


Scrapbook for UML

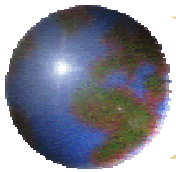
プログラミング
インターフェース

Bash, Perl, Ruby, etc.

外部のスクリプト言語から
SBUMLの各機能を利用可能

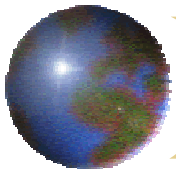


Methods



Methods – User-mode Linux (Intro)

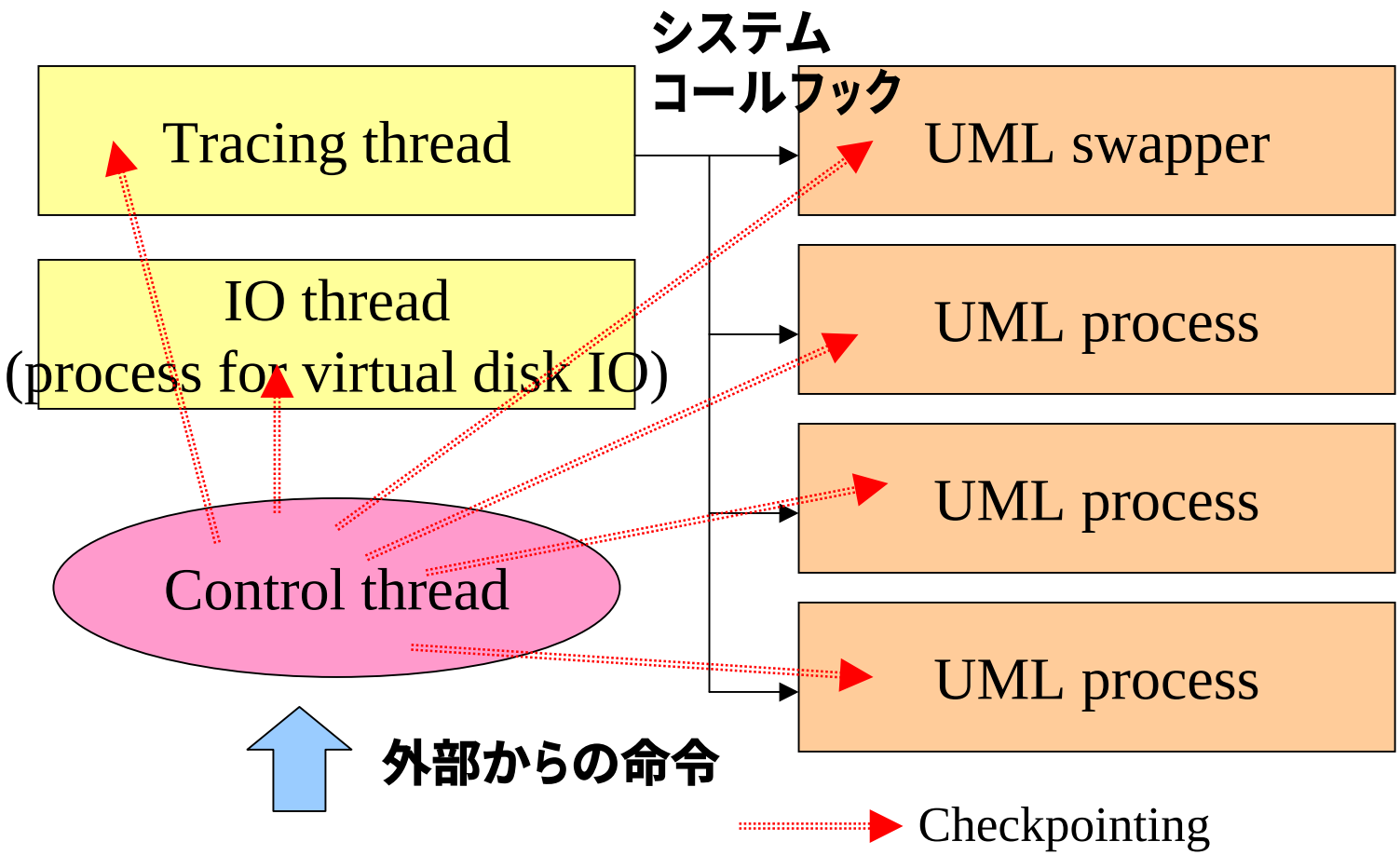
- Jeff Dikeによる**Linuxカーネルへの拡張**
- **Linux内でユーザモードプロセスとして動作するLinuxカーネル**
  - **一種の仮想計算機として動作**
- **参考**URL
  http://user-mode-linux.sourceforge.net/

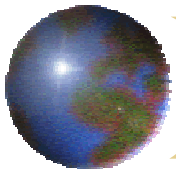


Methods – UML Internals

- 仮想化された実メモリ、仮想ファイルシステムはホストのファイル上に構成される
  - mmap()システムコールによる共有
- 1つのUMLプロセスはホスト側の1ユーザプロセスに対応
- ホスト側のパイプデータ構造を用いたコンテキスト切替
- ptrace()によるシステムコールフック
- SIGUSR1シグナルハンドラによるUMLカーネルモードのエミュレーション

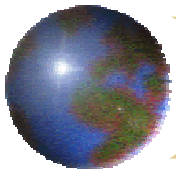
Methods – SBUML Internals





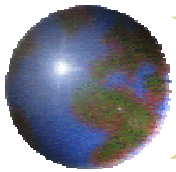
Methods – Checkpointing(1)

- UMLスナップショットの作成に必要な情報
  - 仮想化された実メモリ、仮想ディスク
    - Copy On Writeファイルシステム
  - UMLプロセスアドレス空間のマッピング情報
  - UMLプロセスの実行時コンテキスト情報
  - UMLが使うホスト側ファイルディスクリプタに関する情報

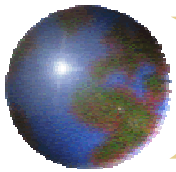


Methods – Checkpointing(2)

- **スナップショットからの再実行時**
  - **ファイルディスクリプタ情報の復元**
  - Tracing thread、control thread**等の特殊なプロセスを先に生成**
  - Control thread**が**UML**プロセスを生成し、各**UML**プロセスは自らのメモリマッピング情報を復元する**
  - **システムコールフックを復元し、保存されたコンテキストから実行を再開する**

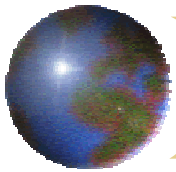


Implementation



Implementation

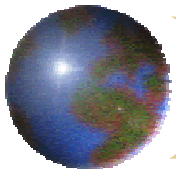
- Linux 2.4.18**及び**UML-patch 2.4.18-36**をベースとして実装**
  - UML**拡張部分に対して約**4000**行の変更及び追加**
- **ホスト**OS (**開発環境**)
  - Linux 2.4.18
  - Pentium4 2GHz, 1GByte RAM



Performance

- スナップショットのファイルサイズ
  (仮想メモリサイズ32MB)
  - 圧縮無し – 約32MB
  - 圧縮有り – 約28MB
  - 差分スナップショット – 約250KB
- 実行時間(スナップショットの圧縮無しの場合)
  - 保存 – 約10秒
  - 再実行 – 約10〜20秒

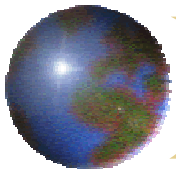
SBUML Demo.



SBUML Demo.

- SBUMLを起動させる
- SBUMLサスペンド機能を用いる
- SBUMLスナップショットを作成する
- スナップショットからの状態復元を行う
  - VNCを用いたデスクトップ環境の復元

SBUML Programming Interfaces

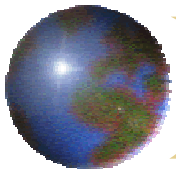


SBUML Programming Interfaces

コマンド名	機能
sbumlfreeze	全UMLプロセスをサスペンドする
sbumlcontinue	全UMLをレジュームする
sbumlsave	スナップショットを保存する
sbumlrestore	スナップショットから再実行する
sbumlanalyzesnapshot	スナップショットから状態抽出する
sbumlschedule	UMLプロセスの優先度を変更する
sbumldonext	状態遷移毎にUMLプロセスを制御する

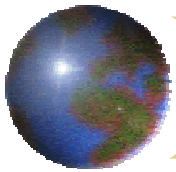
外部からのコマンドラインインターフェースとして提供

Snapshot Programming and SBUML Model Checker



Software Model Checker

- **実環境上でプログラムを動作させることによりモデル検査を行うシステム**
- **非決定性を持つ並行プログラム（マルチプロセス、マルチスレッド）において、起こりうる状態を網羅させる手法を取る**
- **状態数の爆発が問題点**
  - Partial order reduction, data abstraction **等の手法で状態数を減らす解決策**

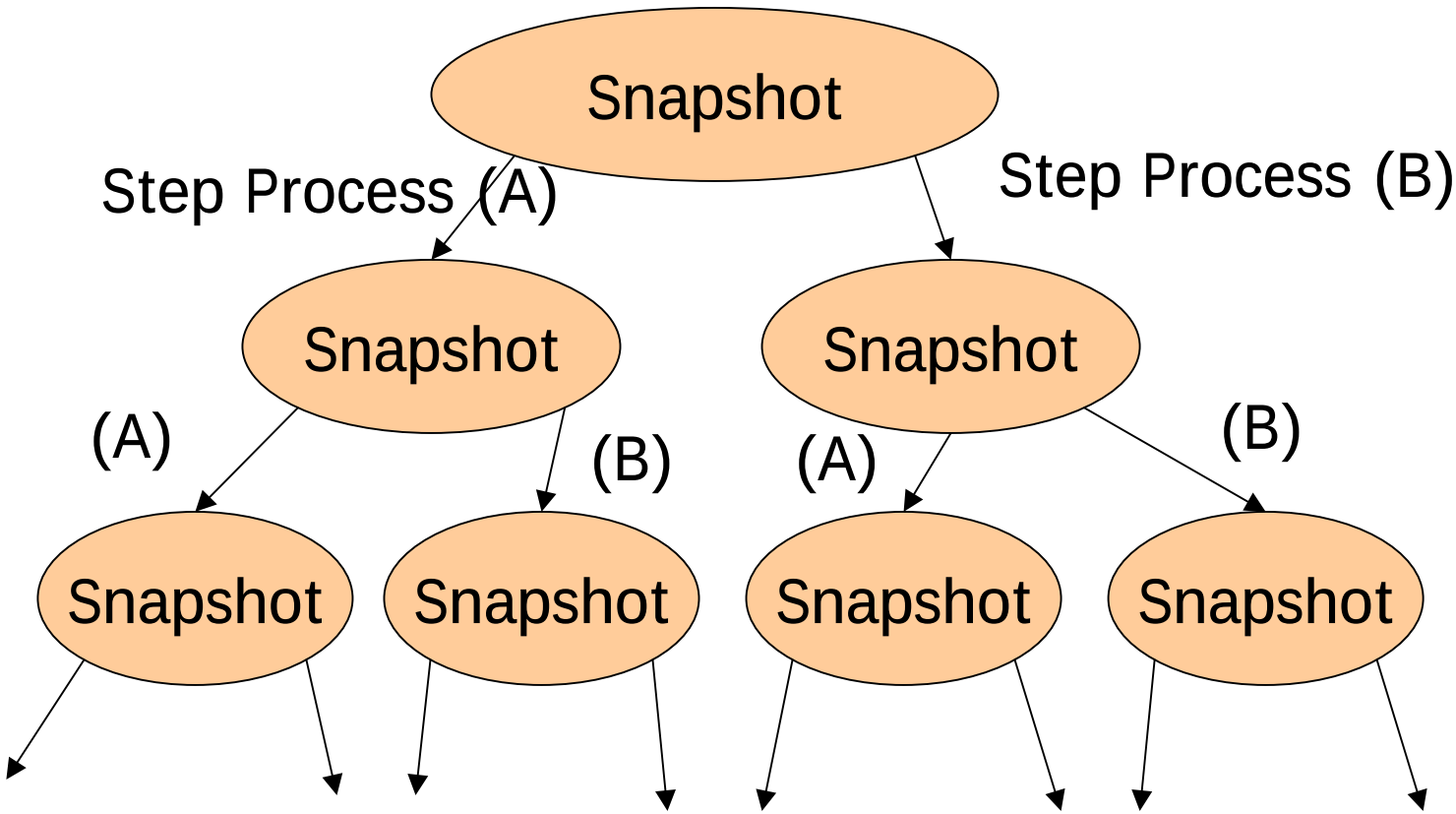


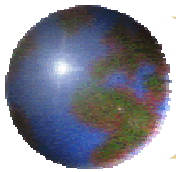
Additional Functionalities

- SBUMLスナップショットは、仮想メモリのイメージをそのまま含んでいる為、そこから内部情報を抽出することが可能
  →プロセスの状態を表すハッシュ値を生成する関数を定義 (スナップショットの解析)
- UMLプロセスのスケジューラ内優先度を上下させる機能を追加 (UMLプロセスの実行制御)
- これら2つを組み合わせると

  → 1つの状態遷移単位でUMLプロセスを制御可能

Snapshot Enumeration





SBUML Model Checker

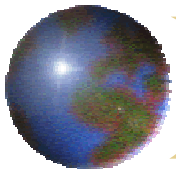
- **前述の追加機能をSBUMLに実装**
- **ソフトウェアモデル検査システムを SBUMLインターフェースを用いて記述**
  - **簡潔なプログラム (Dining Philosopher) を 検証対象とする**
  - **ホスト側のシェルスクリプトから SBUMLインターフェースを利用**
  - **起こりうる状態を列挙し、 デッドロックが起こる可能性を検出**

→SBUMLインターフェースの利用例として紹介

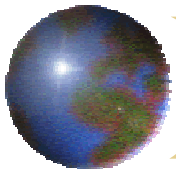
Model Checker Script

```
#!/bin/sh
#
# sbumlenumeratestate
#

mname=$1
hashfile=$2
parentss=${3:-0}
pname=${4:-0}

list=`sbuml--choices $mname`
[ -z "$list" ] && exit

ntime=`date +%s`
sbumlsave $mname $ntime -c –f
OLDIFS=$IFS
IFS=,
```

```
for i in $list ; do
  sleep 1
  sbuml--donext-unless-changing-state ¥
    $mname $i
  hash=`cat sshash.tmp`
  rm -f sshash.tmp
  echo "$parentss", "$pname" > ssinfo

  if grep $hash $hashfile > /dev/null
      then
      exit 0
  else
      echo $hash >> $hashfile
      sbumlenumeratestate ¥
         $mname $hashfile $ntime $i
  fi
  sbumlrestore $mname $ntime -c -f
done
IFS=$OLDIFS
```

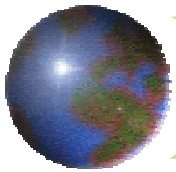


Remaining Work and Conclusion



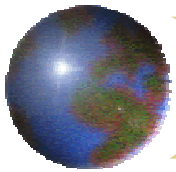
Remaining Work(1)

- SBUMLリリースバージョンの実装
  - パフォーマンス面での最適化
    - より精密なパフォーマンス測定
    - 並行処理によるスナップショット作成
    - スナップショットの圧縮保存
  - 動作の安定化



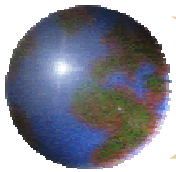
Remaining Work(2)

- SBUMLモデル検査システムの更なる実装
  - 簡潔なテストプログラムの検証には成功
  - より大規模なプログラムの検証を目指す
    - 既存のソフトウェアモデル検査システム(Bandera、Java PathFinder等)で用いられた例
      (RAX, pipeline等)
    - 実際に用いられているサーバアプリケーション
      (nscd, Apache等)
  - Classicalなモデル検査の手法も取り入れる
    (Partial order reduction, プログラムスライシング)



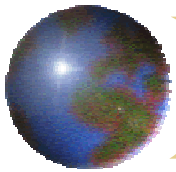
Conclusion

- UML Checkpointingシステムとしての SBUMLの実装
  - リリースバージョンは近日公開予定
- 新たなプログラミング環境としての SBUMLインターフェースの提供
- ソフトウェアモデル検査システムの 新しい実装の第一歩を築く



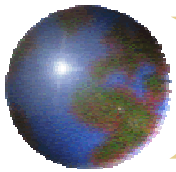
Latest Topics

- **連続した差分スナップショットをFTPサーバを巡回してmergeする機能の実装**
  - **スナップショット配布の際のファイルサイズの問題に対する解決策**
- **状態のハッシュ値を算出する関数をSBUML本体に埋め込む実装**
  - **モデル検査の高速化**
  - **動的ライブラリとしての提供 (実装中)**
- Jeff Dike is coming! (11/25**頃**)



Related Work (VM Checkpointing)

- Network Transferable Computer [Suzaki ’00]
  - Making network-transferable snapshots using VMware, Linux and swsusp
- SoftwarePot [Kato, Oyama ’02]
  - Checkpointing and migrating running software archiving with virtual filesystems



Related Work

- The User-mode Linux Kernel Home Page [http://user-mode-linux.sourceforge.net/]
- Richard Potter and Masami Hagiya, “Computation Scrapbooks for Software Evolution”, Fifth International Workshop on Principles of Software Evolution, IWPSE 2002, 143-147, May 2002.
- M.Musuvathi, D.Park, A.Chou, D.Engler, and D.Dill, “CMC: A Pragmatic Approach to Model Checking Real Code”, Usenix Association, OSDI 2002.