

Pragmatic
Computer Science beyond Theory

情報工学通論

2014/7/8
分散ソフトウェアシステム研究室
福田浩章

目的

- ・ソフトウェアの研究とはどういうものか知る
- ・研究に必要なことは何か知る

Pragmatic
Computer Science beyond Theory 1

研究説明の前に

- ・研究とは？？
 - プログラムを書くこと
 - 論文を読む/書くこと
 - 発表すること
 - 新しい何かをすること
- ・開発との違いは？
 - 新しい概念
 - 新しい手法
 - 性能の改善

Pragmatic
Computer Science beyond Theory 2

研究でよいこと/つらいこと

- ・つらいこと
 - 発表でボコボコにされる
 - 査読で完全否定される
 - 実装が進まない
 - 思った結果が出ない
- ・よいこと
 - うまくいった時の達成感
 - 世界一になれる
 - 発表ついでにちょっと観光

Pragmatic
Computer Science beyond Theory 3

行き先紹介

- ・アメリカ
 - フロリダ、ボストン、ハワイ、シャーロッッピル
- ・ハンガリー
 - ブダペスト
- ・オーストリア
 - インスブルック
- ・スペイン
 - バルセロナ
- ・イタリア
 - ローマ、ラクイラ
- ・ポルトガル
 - フンシャル
- ・フランス
 - パリ、サンマロ
- ・イギリス
 - ランカスター



Pragmatic
Computer Science beyond Theory 4

研究に必要なこと

- ・幅広い知識
- ・鋭い観察力
- ・豊かな想像力
- ・アイデアを実現する技術力（プログラミング能力）
- ・成果物を公表する文書力
- ・英語力
- ・折れない心

Pragmatic
Computer Science beyond Theory 5

研究紹介

- ・ユビキタスコンピューティング
- ・プログラミング言語
- ・ユーザインターフェース
- ・センサーネットワーク
- ・ネットワークセキュリティ
- ・アスペクト指向プログラミング
- ・Rich Internet Application (Web Application)
- ・オペレーティングシステム(maybe 2015~)
- ・Software Defined Network (2014~)

Pragmatic 6

何やってるかわからない？

Pragmatic 7

研究紹介

- ・**ユビキタスコンピューティング**
 - センサーネットワーク
- ・アスペクト指向ソフトウェア工学
 - 非同期プログラミングへの適用
 - ステートフルアспект

Pragmatic 8

ユビキタスとは？

- ・これまで
 - ユーザが特定のコンピュータを利用する
 - ユーザがコンピュータを利用する(主体は人間)
- ・ユビキタス環境
 - 至る所にコンピュータが存在する環境
 - コンピュータがユーザにサービスを提供
 - より便利な世界
 - (人間がダメになるという話も. . .)

Pragmatic 9

具体的には？

Pragmatic 10

ユビキタス環境

Pragmatic 11

実現するには？

- ・ネットワーク
 - WiFi
 - Bluetooth
 - 3G
- ・各種センサー
 - 環境センサー
 - 非接触センサー
 - GPS
- ・賢いプログラム
 - サービスを作り出す

Pragmatic

Wireless Sensor Network(s)

無線センサネットワーク(WSN)とは

The diagram illustrates a Wireless Sensor Network (WSN) architecture. At the top left is a computer monitor labeled "ベースステーション" (Base Station). Below it, several small rectangular modules with antennas represent "センサノード" (Sensor Nodes). Dashed lines connect the base station to these nodes. The nodes are categorized into three distinct groups, each enclosed in a colored oval: a red oval labeled "アプリケーションA" (Application A), a blue oval labeled "アプリケーションB" (Application B), and a green oval labeled "アプリケーションC" (Application C). The word "センサノード" is also written near the nodes.

Pragmatic

WSNの計算資源

- ・個々のノードは電池で稼働

MICAz: WSNの代表的なノード

プログラムメモリ	データメモリ	CPU	電源
128 KB	512 KB	7.4 MHz	単三電池2本

Pragmatic

WSNのアプリケーション

- ・ノードにプログラムを配備することで実現

- ・ノードごとにプログラムを記述しなければならない
- ・WSNの内部構造を考慮してプログラムする必要がある

Pragmatic

WSNのミドルウェア

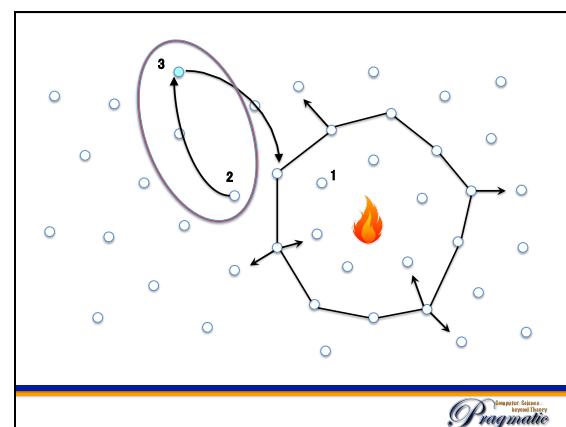
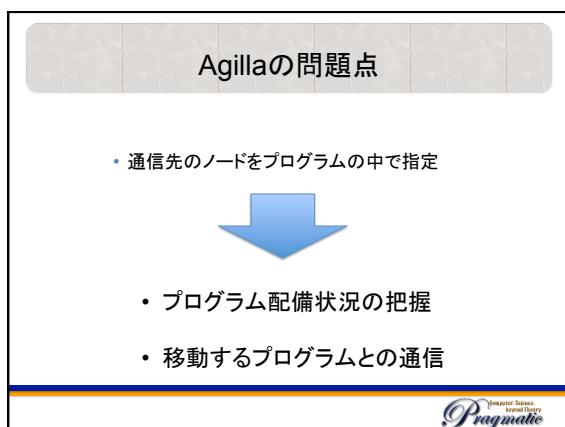
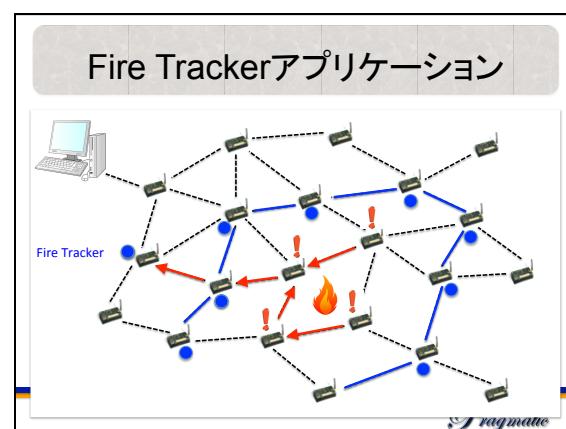
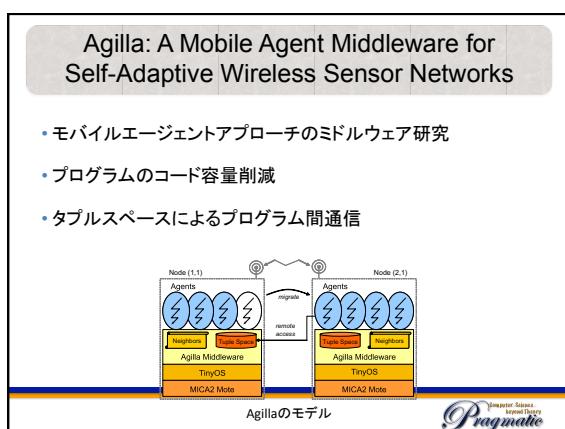
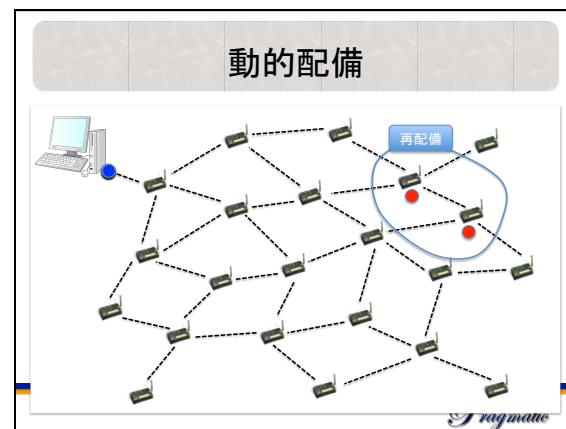
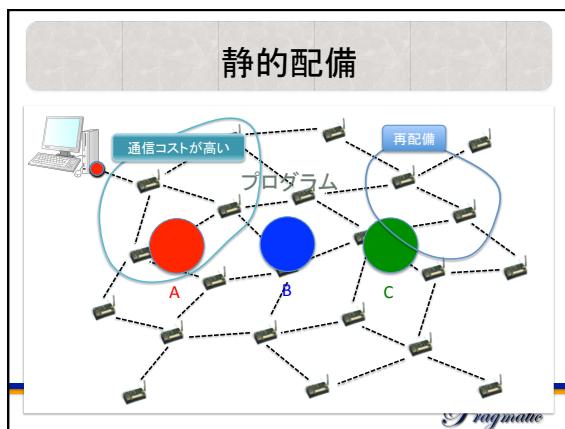
- ・WSNのプログラミングの複雑性を隠蔽
 - (ex. TinyDB , EnviroTrack)
- ・リプログラミングを実現

Pragmatic

リプログラミング

- ・プログラムの変更・追加・削除を容易に
- ・無線によるプログラムの配備
- ・2つのアプローチ
 - ・静的配備
 - ・動的配備

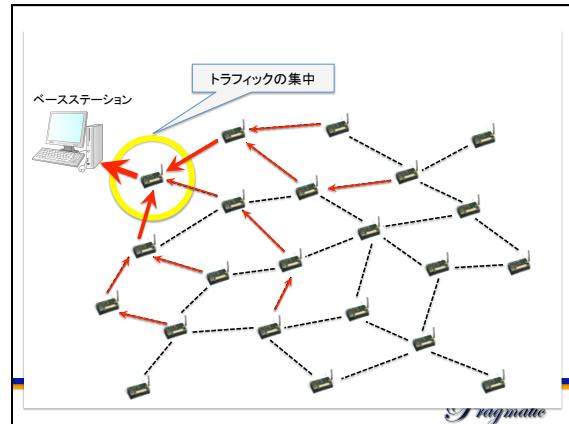
Pragmatic



解決策

- ・プログラムの位置管理を行う機構を実現
- ・分散ハッシュテーブル(DHT:Distributed Hash Table)を利用

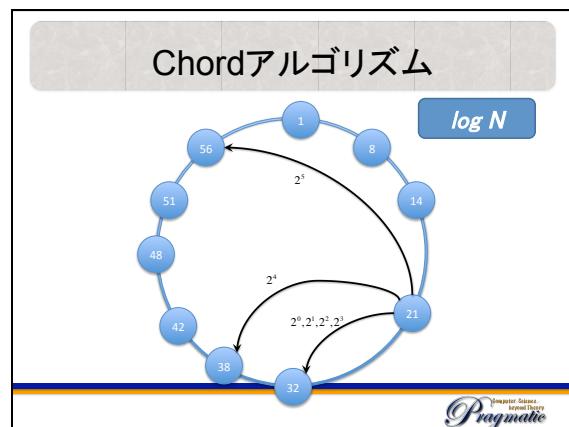
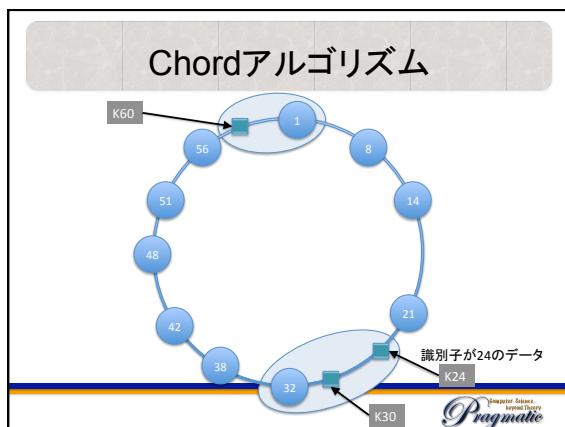
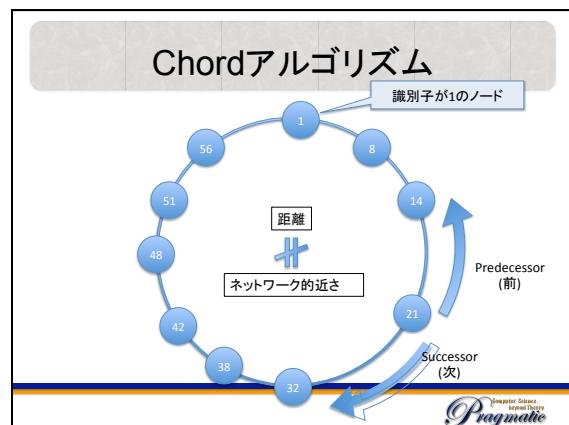
Pragmatic

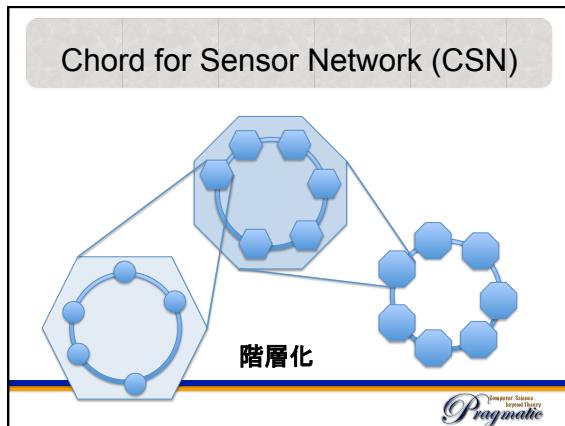


分散ハッシュテーブル(DHT)

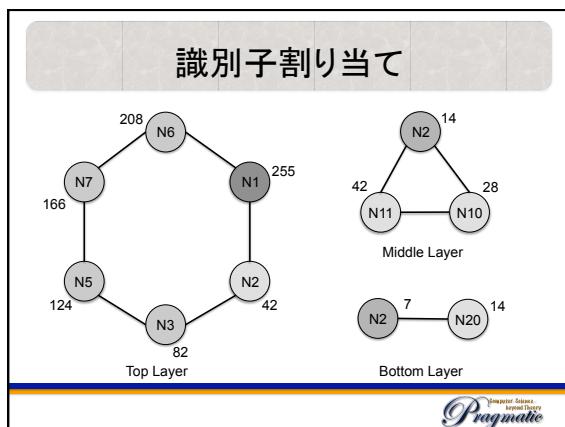
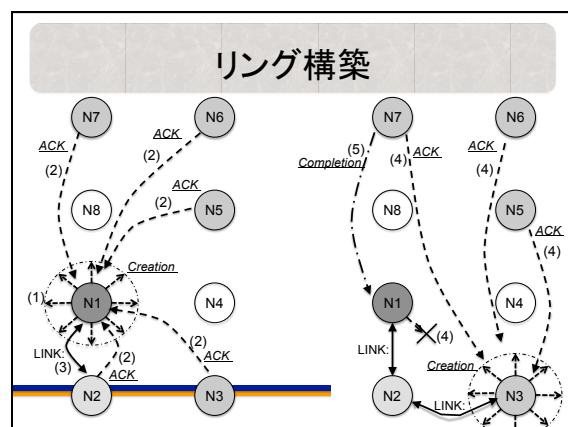
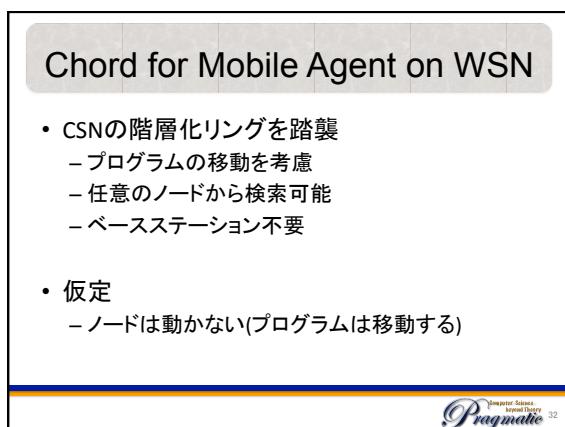
- ・ハッシュテーブルを複数のノードで分割管理する技術
- ・特定のノードに負荷が集中することなく大規模なコンテンツ探索を実現
- ・有限時間内に探索が終了する

Pragmatic





- ### CSNの問題点
- ・プログラムの移動を考慮しない
 - ・検索は常に最上位リングから始まる
 - ・ベースステーションが必要
- Pragmatic Computer Science beyond theory 31



- ### 評価
- ・RandomWalkとCMSNで50回検索実行
 - ・以下を比較
 - 検索時間
 - 検索成功率
 - 消費電力
- Pragmatic Computer Science beyond theory 35

評価環境

Fig. 9 Initial State of structuring DHT.

Fig. 9 初期状態

Fig. 10 After creating a ring of each layer.

Fig. 10 DHT構築完了

Pragmatice 36



検索成功率

手法	検索回数	成功回数	成功率
CMSN	50	50	100 %
Random Walk	50	44	88 %

Pragmatice 38



- ## リング構築の副作用
- 消費電力
 - 18回の検索で逆転
 - 探索時間
 - リング構築に97秒
 - 1,2年の運用を考慮すると十分小さい
- Pragmatice 40

- ## まとめと今後の課題
- プログラムの移動を考慮した位置管理機構の提案
 - 消費電路の削減
 - 探索時間の削減
 - 探索成功率の向上
 - クラスタヘッドのローテーション
 - 一定の割合でノードをスリープ
 - 全体の消費電力向上につながる
- Pragmatice 41

投稿中中(hope to be accepted)

WSNにおける効率的なエージェント位置管理機能

研究紹介

- ・ユビキタスコンピューティング
 - センサーネットワーク
- ・アスペクト指向ソフトウェア工学
 - 非同期プログラミングへの適用
 - ステートフルアスペクト

ソフトウェア工学

- ・一言で言うと. . .
- ・ソフトウェアを効率よく開発するための方法論
 - コンパイラ
 - オブジェクト指向
 - モデリング (ex. UML)
 - モデル検査
 - 要求工学
 - アスペクト指向

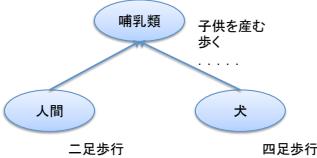
要求の理解の難しさ

オブジェクト指向

- モジュール化して開発効率を高める
 - 繙承
 - カプセル化
 - 多相性
- そもそもなぜ必要?
 - ソフトウェアの大規模化
 - 複数人での開発
 - 過去の資産を有効活用

継承とは

- 親の性質を子に引き継ぐ



Pragmatic 48

多相性とカプセル化

- 多相性
 - 個々のモジュールに異なる振る舞いを許す
 - 人間は二足歩行、犬は四足歩行
- カプセル化
 - 情報を隠蔽する
 - 第三者に公開する情報を限定

Pragmatic 49

ソフトウェアでは(継承)

```

Class Mammalia {
protected heart;
public int heartCount(){....}
public void walk() {...}
}

Class Human extends Honyurui {
public void walk() {
int count = heartCount();
二足歩行の実装
}
}

Class Dog extends Honyurui {
public void walk() {
int count = heartCount();
四足歩行の実装
}
}

```

Pragmatic 50

ソフトウェアでは(多相性)

```

Class User {
public void doit(int type) {
Mammalia obj = Creator.create(type)
.....
obj.walk();
.....
}
}

Class Creator {
public Mammalia create(int type) {
if(type == Human) {
return new Human();
} else if(type == Dog) {
return new Dog();
}
}
}

```

Pragmatic 51

ソフトウェアでは(カプセル化)

```

Class Mammalia {
protected int count;

public void setHeartCount(int cnt) {
if(cnt > 60) {
this.count = 60;
} else {
this.count = count;
}
}

public int heartCount() {
return this.count;
}
}

```

Pragmatic 52

アスペクト指向？

Pragmatic 53

例えは建築: 設備は部屋を横断する

- ・設備ごとに別々の図面

Pragmatic 54

オブジェクト指向とは

- ・抽象化
 - 繙承
 - 多相性
 - カプセル化
- ・モジュール化
 - クラス
- ・モジュールを横断する事柄

Pragmatic 55

アスペクト指向プログラミング

- ・モジュール化技術(OOP)ではうまくまとめられないコードの断片をモジュールとして扱う
- ・対象を本質的な関心事と横断的関心事の集合として捉え、結合ルールで1つのプログラムにまとめる上級のプログラミング手法

Pragmatic 58

アスペクト指向プログラムの構造

- ・モジュール単位「アスペクト」の追加
 - 従来のモジュール単位を横断する処理をアスペクトに書く
 - 従来の単位もそのまま存続
 - 既存のモジュール群へ修正なしにアスペクトを合成
- ・アスペクトのコンパイルイメージ

Pragmatic 57

AspectJ – AOPのデファクトスタンダード

- ・JavaベースのAOP言語
- ・アスペクト定義の拡張
 - ジョインポイント
 - ポイントカット
 - アドバイス

Pragmatic 58

アスペクトの例

クラス	アスペクト
<pre>public class Sample { public void doIt() { Test t = new Test(); t.hello(); } }</pre>	<pre>Public aspect LogAspect{ pointcut main() : call(void Test.hello()); before(): main() { System.out.println("before"); } after(): main() { System.out.println("after"); } }</pre>
	Weave(ウィープ)

Pragmatic 59

AOPの利点

- 横断的関心事が「散らばらない」「もつれ合わない」
 - アプリケーションの機能と非機能の分離が可能
 - モジュール間の結びつきを弱くする:何を,いつ,どのように呼ぶ側のモジュールを再利用可能

Pragmatic 60

AOPの活用シーン

- 横断的関心事のモジュール化と集中管理
 - デバッグ支援:ロギング,トレース,プロファイル
 - 高品質化:キャッシュ,分散,認証
 - ミドルウェアとの疎な結合:トランザクション,永続化
- プロダクトライン開発
 - モジュール疎結合
 - 同一のプログラムセットからシリーズ展開

Pragmatic 61

AOPの問題点

- うまく使わないと複雑になる
 - 保守性低下:全体の見通し, デバッグ困難
 - 効率性低下:
 - 信頼性低下:アスペクト間の干渉
- 従来とは異なるパラダイム:ラーニングコスト
 - 統合開発環境によるサポート
- そもそも何をアスペクトにするか?
 - アスペクト要求工学

Pragmatic 62

アスペクトをRIAに適用(終了)

- RIA(リッチインターネットアプリケーション)
 - Ajax(Aynchronous Javascript and XML)
 - Flash, Flex
 - Silverlight
- 振る舞い
 - イベントドリブン
 - 非同期処理
- 開発
 - デザイナと開発者の協業

Pragmatic 63

成果

- 福田 浩堂, 山本 喜一, モジュールの独立性を考慮したRIA開発フレームワーク, ソフトウェア科学会論文誌(コンピュータリソリューション), Vol. 27, No.4, pp. 97-113, Nov. 2010
- 福田 浩堂, 山本 喜一, AspectFX:アスペクト指向によるRIA開発での協業を支援するフレームワーク, 電子情報通信学会論文誌, Vol. J93-D, No.7, pp. 1165-1178, Jul. 2010
- 福田 浩堂, 山本 喜一, RIAにおけるデザイナーと開発者の完全分業を実現するフレームワーク, 情報システム学会論文誌, Vol. 4, No.2 pp.28-43, Sep. 2008.
- Hiroaki Fukuda, Yoshikazu Yamamoto, "A system for supporting development of large scaled Rich Internet Applications, In Proceedings of 23th IEEE/ACM International Conference on Automated Software Engineering 2008 (23% acceptance rate).
- Hiroaki Fukuda, Yoshikazu Yamamoto, A Framework for realizing complete separation of developer's and designer's work in Rich Internet Application, In Proceedings of 10th International Conference on Enterprise Information Systems, Vol. 2, pp. 137-142 Barcelona, Spain, June 2008(below 10% acceptance rate)

Pragmatic 64

成果

- Hiroaki Fukuda, Yoshikazu Yamamoto, A Framework for realizing complete separation of developer's and designer's work in Rich Internet Application, In Proceedings of 10th International Conference on Enterprise Information Systems, Vol. 2, pp. 137-142 Barcelona, Spain, June 2008(below 10% acceptance rate)
- Hiroaki Fukuda, Yoshikazu Yamamoto, "Yet another aspect in Rich Internet Applications", In Proceedings of Asian Workshop on Aspect-Oriented Software Development(AOAsia 2009), pp. 17-24, Auckland, New Zealand, November 16-20 2009.
- Hiroaki Fukuda, Yoshikazu Yamamoto, "Modularity oriented framework for Rich Internet Application", In Proceedings of IASTED International Conference on Software Engineering and Applications(SEA 2009), pp. 92-99, Cambridge, Massachusetts, USA, November 2-4, 2009.
- Hiroaki Fukuda, "AOP based language extension for web development", In proceedings of the 12th IASTED International Conference on Software Engineering, pp. 744-751, Innsbruck, Austria, Feb. 11-13, 2013.
- Kohei Nagashima, Hiroaki Fukuda, and Shingo Takada "Aspect-jQuery: An Aspect-Oriented Framework for jQuery", to appear in proceedings of AOAsia 2013 Fukuoka, Japan, Mar. 24, 2013.

Pragmatic 65

そして...
非同期プログラミングへ

Pragmatic 66

Synchronous VS Asynchronous

- Sync(Blocking)


```
var x = search(url);
display(x);
```

Block/Stop until "search" returns
- Async(Non Blocking)


```
var x = search(url);
display(x);
```

Do NOT block/stop until "search" returns

```
display(NULL);
```

Pragmatic 67

Callback Solution

- Callback Function


```
search(url, searchComplete);
```

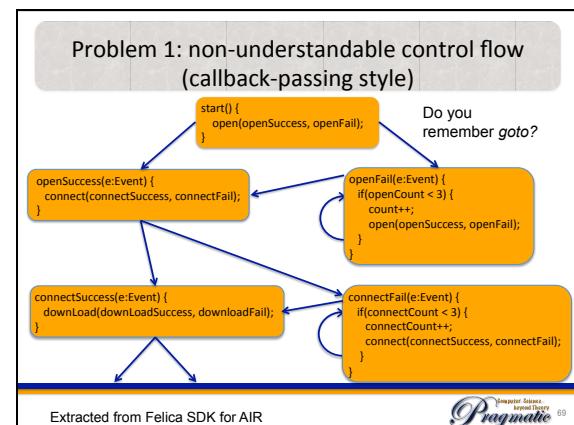
call

 - Call Function with callback function
 - Callback is called when an async process is done using new function: searchComplete
 - Get a result by using arguments

```
function searchComplete(e:Event) {
  var x = e.x;
  display(x);
}
```

callback

Pragmatic 68



Problem 2: avoiding callback-passing style but no modular implementation

```
fs.readdir(source, function(err, files) {
  if (err) {
    console.log('Error finding files: ' + err)
  } else {
    files.forEach(function(filename, fileIndex) {
      console.log(filename);
      gmsource + filename).size(function(err, values) {
        if (err) {
          console.log('Error identifying file size: ' + err)
        } else {
          console.log(filename + ':' + values)
          widths.forEach(function(width, index) {
            height = Math.min(values['height'], width);
            console.log('resizing ' + filename + ' to ' + height + 'x' + height)
            this.resize(width, height).write(destination + 'w' + width + '-' + filename, function(err) {
              if (err) console.log('Error writing file: ' + err)
            })
          }, bind(this))
        }
      })
    })
  }
})
```

e.g.
http://callbackhell.com/

Pragmatic 70

Existing Proposals (1/2): An Emulation of a Thread Library

```
function click():void {
  search(url); // async
  next(searchComplete);
  error(Error, searchError);
}

function searchComplete():void {
  if(photoDisplay != null) {
    photoDisplay.stopAnimation(); // async
    next(display);
  } else {
    display();
  }
}

function display():void {
  var x = searchResult; // global obj
  photoDisplay = new PhotoDisplay(x);
  photoDisplay.start();
}
```

Explicit specification of the async control flow
We need a global object to pass information

This is still callback-passing style

Extracted from FlickrSphere

Pragmatic 71

Existing Proposals (2/2): Fluent Interface for callback-passing style

`search(url).then(display).catch(error);`

Force the async control flow

`function search(url){
 var d = new Deferred(); // Object in Library
 urlSearch.addEventListener(Complete,
 function(e:Event):void {
 if(e.target != null ?
 d.then(e.target) :
 d.catch(e.target))
 ifThen(display, func)
 }
);
}`

Corresponds to "searchComplete"

`search(url)
 then(display)
 catch(error);
 ifThen(display, func)`

A crosscutting concern !!, because search needs to insert the library calls
Many operators may be required

Pragmatic 72

Summarize

- Callback Solution
 - Modular but non understandable the async control flow
 - Understandable async control flow but it is not modular
- Existing Proposals
 - Improve the understanding of the async control flow
 - But:
 - Thread:
 - Require an explicit specification of the async control flow
 - Require a global object to pass information
 - Fluent Interface:
 - Add a crosscutting concern (similar to observer pattern)
 - Many operators may be required

Pragmatic 73

Solution and Research Question

- The synchronous allows developers to improve the understanding the execution of a program
- How to write an async control flow as a sync fashion?

SyncAS

Pragmatic 74

What is SyncAS?

- A library which controls the async control flows
- Control to:
 - (Virtually) block the execution at arbitrary points
 - Restart the execution
- How to specify the block points?
 - Using aspects (AOP)

Pragmatic 75

Async vs SyncAS

`var sm = new SearchManager();
sm.search(url, searchComplete);`

`function searchComplete(e:Event):void {
 searchResult = e.x; // global object
 stopAnimation(display);
}`

`Function display():void {
 var x = _searchResult;
 var photoDisplay = new PhotoDisplay(x);
 photoDisplay.display();
}`

SyncAS
It does not need callback-passing style

Pragmatic 76

How SyncAS works

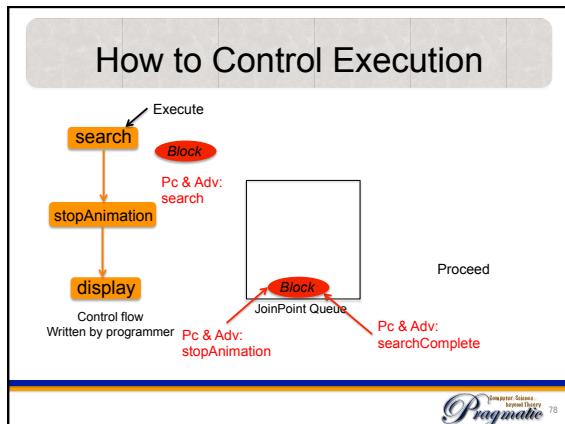
Q. How to specify these "Blocks" ?

`var sm = new SearchManager();
var x = sm.search(url);
stopAnimation();
var pd = new PhotoDisplay(x);
pd.display();`

Block
Restart
Block
Restart

A. Aspect Pointcut & Advice

Pragmatic 77



Pointcut and Advice for Blocking

```
var pc:Function = function(jp:JoinPoint, env:Environment) {
    return (jp.isCall() &&
        (jp.getClassName() == "SearchManager") &&
        (jp.getMethodName() == "search"));
}

var adv:Function = function(jp:JoinPoint, env:Environment) {
    var sc:SeqControl = env.get("seqcontrol") as SeqControl;
    sc.addToggle("block");
}

aspect.add(pc, adv, Aspect.AFTER);

AspectRuntime.deploy(aspect);
```

Pointcut and Advice for Restarting

```
var pc:Function = function(jp:JoinPoint, env:Environment) {
    return (jp.isExec() &&
        (jp.getClassName() == "SearchManager") &&
        (jp.getMethodName() == "searchComplete"));
}

var adv:Function = function(jp:JoinPoint, env:Environment) {
    AspectRuntime.releaseToggle("block");
}

aspect.add(pc, adv, Aspect.AFTER);

AspectRuntime.deploy(aspect);
```

Entire pc/adv for control sequence

```
var aspect:Aspect = new Aspect();

pc / adv pair for blocking as pc1 / adv1
aspect.add(pc1, adv1, Aspect.AFTER);

pc / adv pair for restarting as pc2 / adv2
Aspect.add(pc2, adv2, Aspect.AFTER);

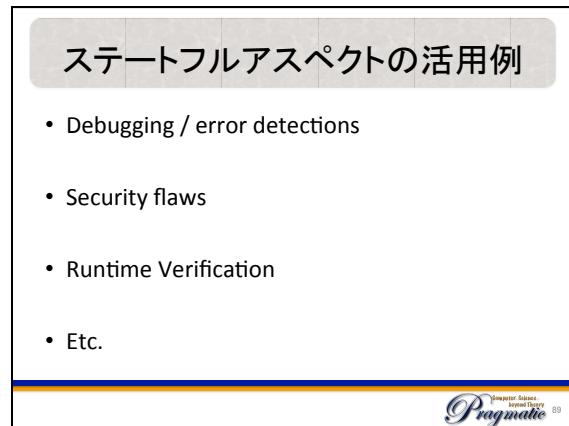
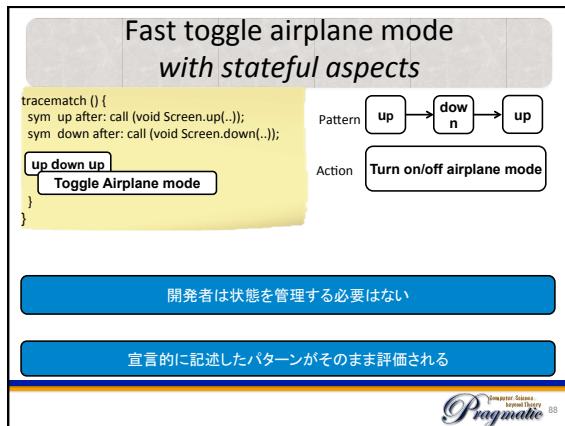
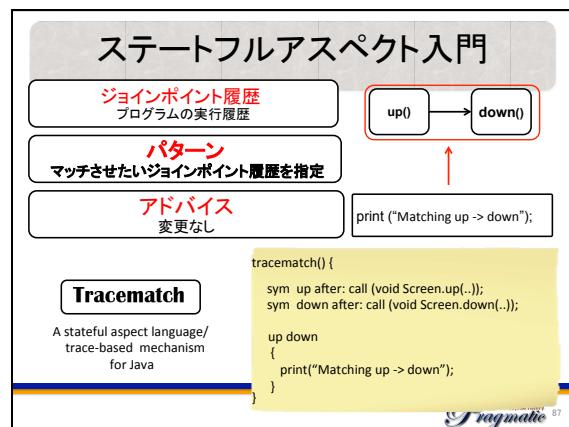
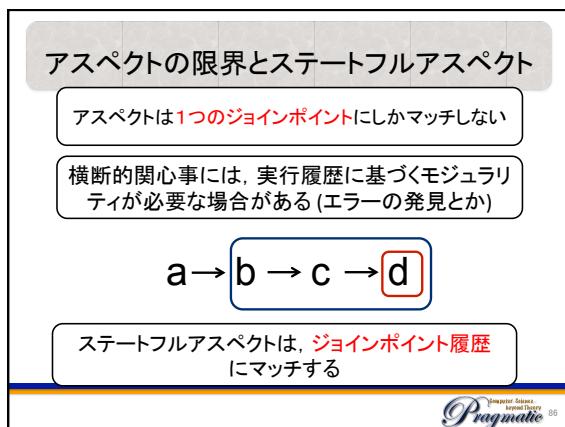
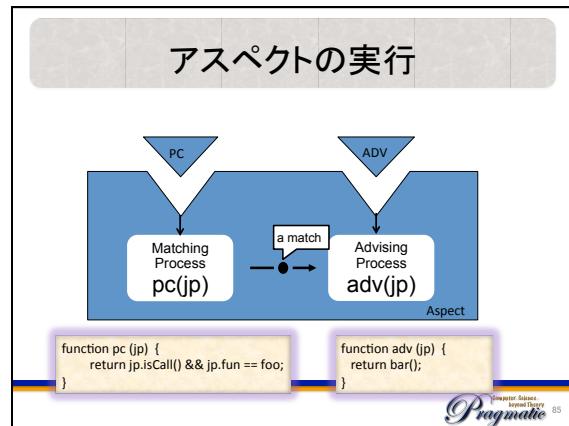
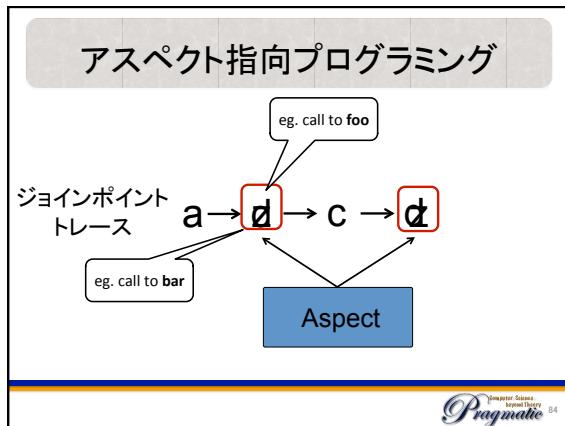
AspectRuntime.deploy(aspect);
```

- ### 現在とこれから
- 現在の状況
 - ランタイムの実装完了
 - 論文の着手
 - これから
 - トランスレータの実装
 - 別のアプリケーションに適用
 - 別の言語(e.g. Dart)に適用

超最近の話題

ステートフルアスペクト

An Expressive Stateful Aspect Language



マッチングプロセスの複雑さ

パターン $a \rightarrow b$

ジョインポイント履歴 $a \rightarrow a \rightarrow b$

ステートフルアスペクトのマッチング処理はより複雑になる

Pragmatic 90

アドバイス実行の複雑さ

パターン $a^{v=?} \rightarrow b$

ジョインポイント履歴 $a^{v=1} \rightarrow a^{v=2} \rightarrow b$

ステートフルアスペクトのアドバイス実行はより複雑になる

Pragmatic 91

ステートフルアスペクトの実行

- マッチングプロセスではパターンに何度もマッチする
- アドバイスの実行は一度に同時に起こる

Pragmatic

Tracematchesの問題点

Pattern: $up \rightarrow down \rightarrow up$

Join Point Trace: $up \rightarrow down \rightarrow up \rightarrow toggle \rightarrow up$

```
tracematch () {
    sym up after: call (void Screen.up(..));
    sym down after: call (void Screen.down(..));
    sym toggle after:
        call(void *.toggleAirplaneMode(..));
    up down up {
        toggleAirplaneMode();
    }
}
```

Tracematches 複数マッチをサポートする
しかし、複数マッチの意味を変えることはできない
開発者は意図的にシンボルを挿入し、思い通りに動作させる必要がある

Pragmatic 93

その他のステートフルアスペクト言語 (e.g. EventJava, Alpha, and Halo)

Pattern: $up \rightarrow down \rightarrow up$

Join Point Trace: $up \rightarrow down \rightarrow up \rightarrow down \rightarrow up$

airplane modeがon/offされるタイミングを知るのが困難

ステートフルアスペクト言語は意味論を変更する機構を有していない

Pragmatic 94

ステートフルアスペクト言語の例

- Tracematch
- HALO
- Alpha
- Java-Mop
- EventJava
- Tracecuts
- PQL & PQTL

すべての言語で異なる意味論、パターン言語を持つ
理由: 扱う問題領域の相違

表現力豊かなステートフルアスペクト言語の提案
既存言語の意味論を扱うことができ、更に新しい意味論の定義が可能

Pragmatic 95

