

.NET to .NET マイグレーション ～ 移行計画立案と実践 ～

日本ユニシス株式会社
.NETセンター .NET技術検証室
豊田裕之 / 猪股健太郎

Microsoft®
tech.ed
2008

Agenda

- ▶ 本セッションの背景
- ▶ 本セッションの概要
- ▶ 日本ユニシスの Windows/.NET への取り組み
- ▶ 移行計画立案のための前提知識
- ▶ 移行計画立案
- ▶ Windows 基盤移行の注意点

本セッションの背景

Microsoft .NET Framework 1.1/3.5 移行ホワイトペーパー
(近日公開予定)

.NET Framework プログラム移行関連

T2-202 .NET to .NET マイグレーション
～ .NET Framework 1.x から最新環境への移行のために ～

TH-306 .NET to .NET マイグレーション
～ 移行計画立案と実践 ～ (本セッション)

それ以外 + α



本セッションの背景

- ▶ .NET Framework の移行において、アプリケーションの移行だけを考えれば、システムの移行ができますか？

No!

- ▶ 新機能追加、品質保証(テスト戦略)...
- ▶ Windows、サーバー、クライアント、ネットワーク機器などの更改・移行... など

移行計画立案段階で決めておかなければ
ならないことは多々ある！

セッション概要

▶ 想定受講者

- ▶ .NET to .NET 移行プロジェクトの計画立案者
- ▶ .NET to .NET 移行プロジェクトのアーキテクト

▶ 内容

- ▶ .NET to .NET 移行プロジェクトの計画立案にあたり知っておくべきこと。
- ▶ .NET Framework 上のアプリケーション移行のプログラマー向けの内容は深く言及しない。
- ▶ 関連する Windows 基盤技術については言及する。

日本ユニシスの Windows/.NET への取り組み

【お問い合わせ】

日本ユニシス株式会社

.NETセンター

<http://www.unisys.co.jp/dotnet/>

[Mailto:dotnet-box@unisys.co.jp](mailto:dotnet-box@unisys.co.jp)

2008

Windows Server® 2008移行ユーザー向けにシステム構築サービスを提供開始

2007

百五銀行、日本ユニシス、マイクロソフト 世界初、Windows Server、SQL Server を基盤としたフルバンキングシステムが百五銀行で稼働開始

2006

SharePoint Server 2007、Exchange Server 2007の構築サービス提供開始
開発方法「LUCINA® for .NET」をマイクロソフトの最新の開発ツール
(Visual Studio Team System、Visual Studio Team Foundation Server)に対応
日本スポーツ振興センターのWindows大規模システムtotoシステムが稼働開始

2005

.NET対応のミッションクリティカル構築・運用支援ミドルウェア
「MIDMOST for .NET」販売開始

2004

セブン銀行の勘定系システムをWindowsベースで刷新(国内初の取り組み)
アサヒビールグループの戦略的DWH構築(64bit Windows + SQL Server 2000)

2003

日本ユニシス、百五銀行の次期基幹系システムをWindowsベースのオープンシステムで共同開発

2002

.NETベースのアプリケーション構築支援開発方法「LUCINA® for .NET」を提供開始
.NETビジネス専任組織「.NETビジネスディベロプメント」を新設

2000

日本ユニシスとマイクロソフト、新世代データセンターシステム分野で包括提携
最大32CPU搭載可能な「Unisys e-@ction Enterprise Server ES7000」を販売開始



移行計画立案のための前提 知識

Microsoft®
tech.ed
2008

.NET Framework バージョンの 関係

.NET Framework 3.5

ASP.NET AJAX

LINQ

etc...

.NET 3.xは
.NET 2.0上で
動作する

.NET Framework 3.0

Windows Presentation Foundation (WPF)

Windows Workflow Foundation (WF)

Windows Communication Foundation (WCF)

Windows CardSpace

.NET Framework 2.0

基本クラス ライブラリ

ASP.NET

ADO.NET

Windows Form

Visual Studio と .NET Framework の関係

Visual Studio	生成するソースコード
Visual Studio.NET(2002)	.NET Framework 1.0
Visual Studio.NET 2003	.NET Framework 1.1
Visual Studio 2005	.NET Framework 2.0 (VS 2005 用)
Visual Studio 2008	.NET Framework 2.0 (VS 2008 用)
	.NET Framework 3.0
	.NET Framework 3.5

- ▶ Visual Studio 2008 でターゲットを .NET Framework 2.0 にしても、Visual Studio 2005でもビルド可能なソース・コードを生成しないことに注意。

チーム開発において、Visual Studio 2005/2008はいずれかに統一

MS製品のサポートについて

▶ 2種類のサポート期間

▶ メインストリーム・サポート(5年)

- ▶ 修正パッチ、セキュリティ・パッチがリリースされる

▶ 延長サポート(5年)

- ▶ セキュリティ・パッチだけがリリースされる
- ▶ 修正パッチは出ない！

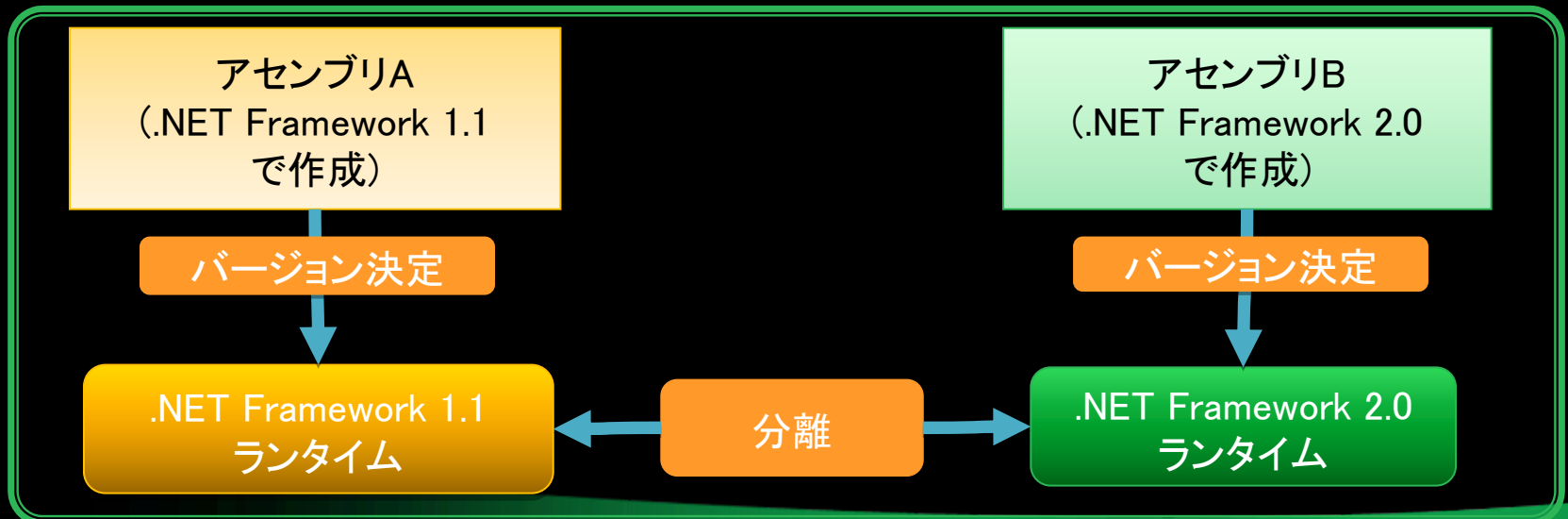
▶ サポート ライフサイクル ポリシー

- ▶ [http://support.microsoft.com/default.aspx?scid=fh;\[ln\];lifecycle](http://support.microsoft.com/default.aspx?scid=fh;[ln];lifecycle)

要求される品質レベル、改修頻度により、延長サポートでも良いか決定

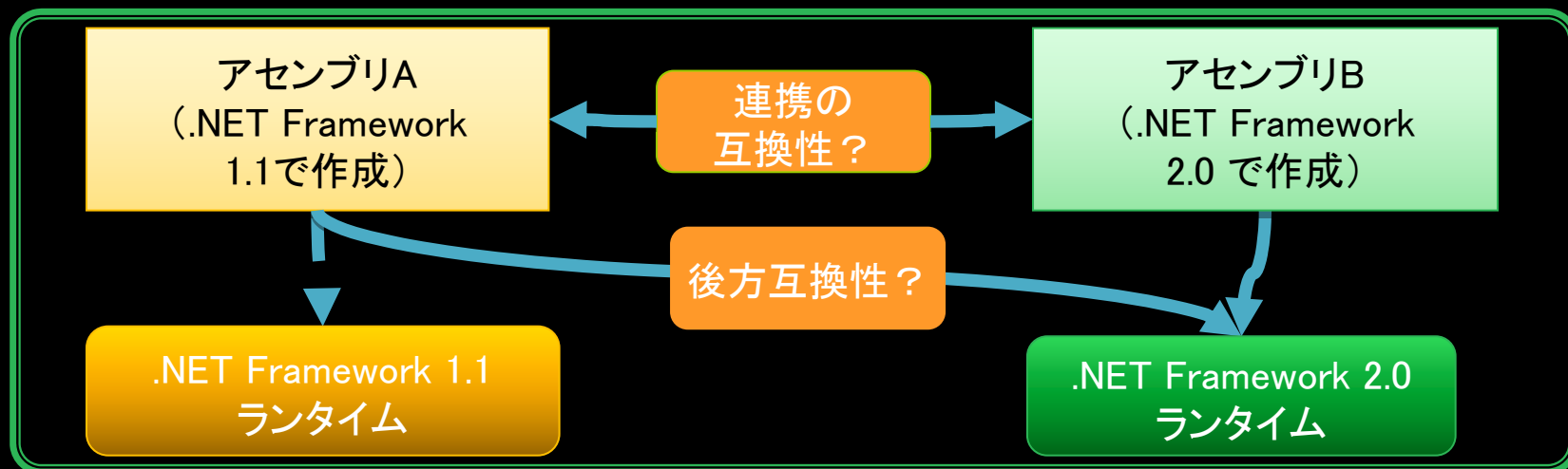
サイドバイサイド実行 (1)

- ▶ .NET Framework は、1システム内で複数のランタイムが互いに影響を与えずに動作可能
- ▶ アセンブリはビルドされた時の .NET Framework バージョンと同じランタイムで動作



サイドバイサイド実行 (2)

- ▶ .NET Framework 1.1 アセンブリを .NET Framework 2.0 ランタイムの下位互換機能で動作させたり、バージョンの異なるアセンブリを連携させることが必要になることもある。



サイドバイサイド実行については、「Microsoft .NET Framework 1.1/3.5 移行 ホワイトペーパー」に注意点を解説。

移行計画立案

Microsoft®
tech.ed
2008

事前調査

- ▶ 移行対象システムに仕様変更点に該当するか機能があるか調査
 - ▶ .NET Framework 2.0 での重大な変更点
 - ▶ <http://www.microsoft.com/japan/msdn/netframework/programming/breakingchanges/default.aspx>
 - ▶ WSE(Web Services Enhancements) や 3rd. パーティ製部品を利用している場合、同様に調査

一般的な業務システムでは、.NET Framework 1.1/2.0 の仕様変更点に該当する項目は少ないと考えられる。

事前検証 (1)

- ▶ 移行対象システムのソリューションファイル形式の入手
- ▶ ビルド試行環境の構築
 - ▶ Visual Studio 2008 および 3rd. パーティ製部品など
- ▶ ソリューション変換作業 (Visual Studio の移行ウィザードを利用)
- ▶ (事前調査で該当した).NET Framework の仕様変更点に対するコード修正

ソリューション変換作業については、「Microsoft .NET Framework 1.1/3.5 移行ホワイトペーパー」に注意点を解説。

事前検証 (2)

- ▶ 一括リビルドテスト
- ▶ ビルドエラーおよび警告に対するコード修正
- ▶ 実行試行環境の構築、アセンブリの配置
- ▶ ランタイムエラーテスト
- ▶ 残タスクのリストアップ
 - ▶ プロジェクト変換エラー、ビルドエラー、ランタイムエラー発生箇所に対する修正

(弊社事例では)ソリューション変換～リビルドの工数は非常に少ない。

移行計画立案(1)

- ▶ 移行目的、制約事項と優先度の明確化
 - ▶ 新機能追加による新たなビジネス価値の付加？
- ▶ システムのライフサイクル、システム資産の調査、維持・保守計画の立案
 - ▶ システム全体のライフサイクルに応じて、ハードウェア更改、OS やミドルウェアのバージョンアップも検討する。

単純な移行では、システムのビジネス価値は向上しない。

アプリケーション移行だけでなく、システムのライフサイクルを検討。アプリケーションと基盤の移行を同時に行うことで、大きなコストを占めるテスト工数の「2重払い」をしないことが可能か検討。

移行計画立案 (2)

▶ 移行方式検討

▶ アーキテクチャ変更の検討

- ▶ ソリューション変換ウィザードでは、.NET Framework の新アーキテクチャへの変換は行われない。
 - ▶ たとえば、DataAdapter から TableAdapter に変換されることはない。

▶ 一括移行 / 段階的移行

- ▶ 新機能開発分、機能改修部分のみ新バージョンを用いる場合、サイドバイサイド実行の利用

サイドバイサイド実行には、制限事項があるので注意。
また、実行ランタイムバージョン決定の仕組みの理解が必須。

移行計画立案 (3)

- ▶ 達成すべき品質レベルから、テスト密度と網羅性の検討
 - ▶ 全機能の再テストはコスト負担が大きい。
- ▶ 工数見積もり
 - ▶ 事前調査、事前検証作業により、アプリケーション移行作業の工数には比較的高い精度で見積もりが可能。
 - ▶ テスト工数
 - ▶ 最大は「開発～改修のテスト工数合計」

システム移行においてテスト工数の占める割合は大きい。
システム価値向上とリスクを考え、投資対効果を極大化する。

移行計画立案 (4)

▶ まとめ

▶ 事前調査、事前検証段階

- ▶ 移行のコスト、実現可能性、残タスクの把握

▶ 移行計画立案段階

- ▶ 移行目的、制限事項明確化

- ▶ システム全体のライフサイクル、保守計画検討

- ▶ 移行方式決定

- ▶ アーキテクチャ変更、一括/段階移行

- ▶ 品質目標定義

- ▶ テスト密度、網羅性から投下コストとリスク勘案

- ▶ コスト、スケジュール、体制、リスク管理などの決定

Windows 基盤移行の注意点

Microsoft®
tech.ed
2008

移行における基盤の注意点

- ▶ Windows 基盤の移行に伴い、「強要」される移行
 - ▶ IIS 6.0 to IIS 7.0 (Windows Server 2008)
 - ▶ ユーザーアカウント制御 (UAC) の考慮
- ▶ 新機能のメリットを享受するために行う移行
 - ▶ ノータッチデプロイメント → ClickOnce

Windows Server 2008 移行

IIS 6.0 から IIS 7.0 の移行

Microsoft®
tech.ed
2008

IIS 7.0 への移行検討

Windows OS と IIS のバージョン対応

OS	IIS	mode	.NET Framework 3.0 以上搭載可能
Windows 2000	IIS 5.0		×
Windows XP	IIS 5.1		○
Windows Server 2003	IIS 6.0	プロセス分離モード IIS 5 の分離モード	○
Windows Vista			○
Windows Server 2008	IIS 7.0	クラシックモード 統合モード	○

- ▶ IIS の後方互換性は完璧ではない
- ▶ .NET Framework のバージョン移行と同時に、IIS のバージョン移行も検証する必要がある

IIS 7.0 への移行検討

.NET Framework と IIS

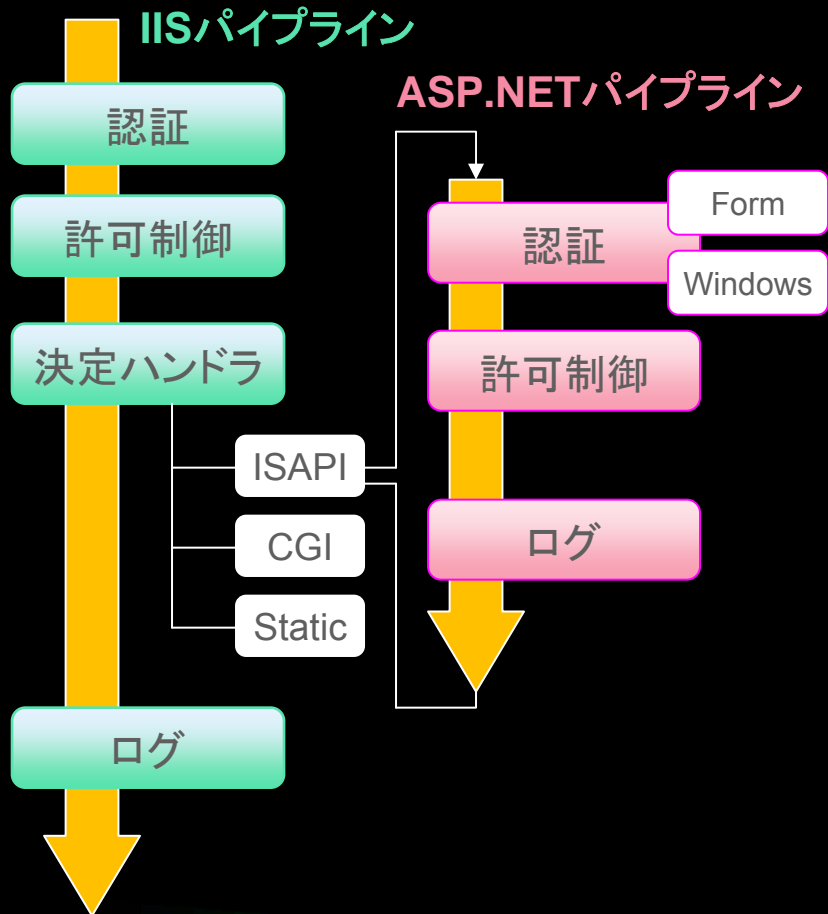
移行後		Windows Server 2008			
		IIS 7.0 クラシックモード		IIS 7.0 統合モード	
移行前		F/W 2.x	F/W 3.x	F/W 2.x	F/W 3.x
	Windows Server 2003	F/W 1.x	複数の移行パターン		
IIS 6.0 プロセス分離モード	F/W 2.x				
(or IIS 5 の分離モード)	F/W 3.x				

- ▶ IIS 7.0 はモードによって内部動作が大きく異なる
- ▶ まずは.NET Framework のバージョンを上げ、アプリケーションの動作確認後に、IIS の移行検証を行うとよい
- ▶ プロセス分離モードで動作できることが大前提

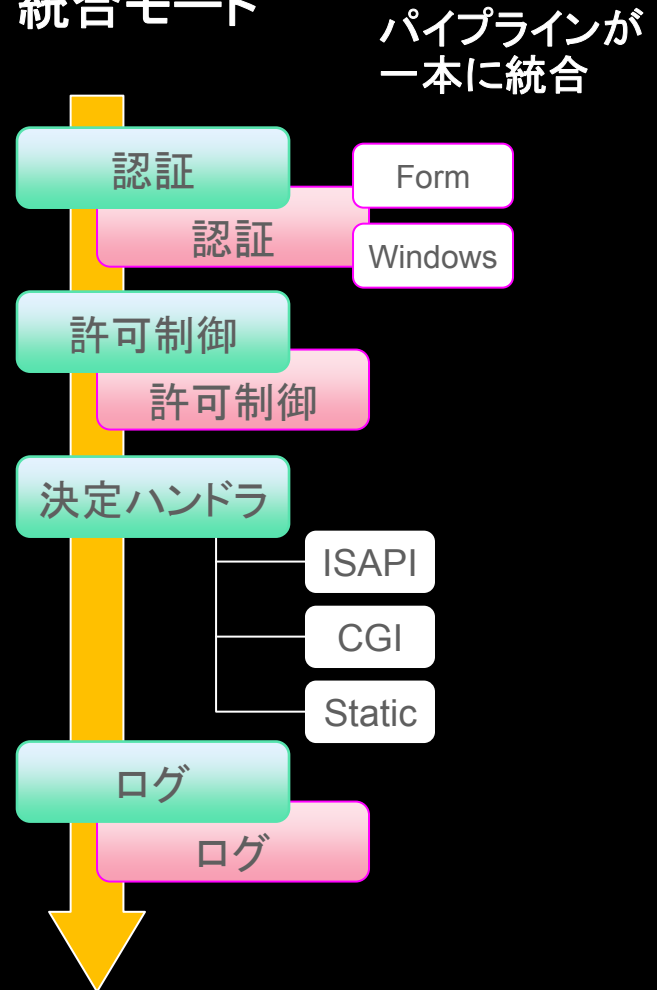
IIS 7.0 クラシックと統合モード

パイプラインアーキテクチャの違い

クラシックモード



統合モード



IIS 7.0 クラシックと統合モード

IIS 6.0 から IIS 7.0 への互換性

- ▶ クラシックモードではほとんどの ASP.NET 2.0 アプリケーションは動作する
- ▶ 統合モードでは検証・修正が必要

説明	クラシック	統合
パスポート認証	移行不可	移行不可
リクエスト URL に符号化されない「+」等の文字列を含む	要修正	要修正
クエリストリングが 2KB を超えるようなリクエストが発生する	要修正	要修正
統合 Windows 認証とフォーム認証の同時使用	移行可	大改修
Application_Start イベントで Context オブジェクトから Request を操作	移行可	移行不可
DefaultHttpHandler または派生したカスタム HTTP ハンドラを使用	移行可	移行不可
カスタムモジュール及びハンドラを実装	移行可	要修正
フォーム認証利用	移行可	構成を確認
クライアント偽装の実装	移行可	構成を確認

(<http://learn.iis.net/page.aspx/381/aspnet-20-breaking-changes-on-iis-70/> より)

IIS 7.0 Key Point

IIS 7.0 の新機能

▶ 運用・管理

- ▶ applicationHost.config による構成情報の一元管理
- ▶ appCmd / WMI / マネージド API 等の豊富な管理ツール
- ▶ 失敗した要求トレース / ETW との統合
- ▶ 機能のモジュール化に伴うセキュリティの向上

▶ クラシック / 統合モードに関係なく利用可能

▶ アプリケーションの拡張性

- ▶ マネージモジュールを IIS のパイプラインに組み込むことが可能
- ▶ フォーム認証結果を拡張子の異なる静的ファイルに対して適用可能

▶ 統合モードでのみ利用可能

IIS 7.0 Key Point

IIS 7.0 の新機能の注意点

▶ 運用・管理

- ▶ applicationHost.config による構成情報の一元管理
- ▶ appCmd / WMI / マネージド API 等の豊富な管理ツール

Caution!

- ・それぞれのツールで設定できるものとできないものがある
- ・機能の委任設定を的確に指定する必要がある
- ・IIS マネージャの挙動によって構成ファイルが壊れる可能性がある
→ applicationHost.config のバックアップを適宜とる

▶ アプリケーションの拡張性

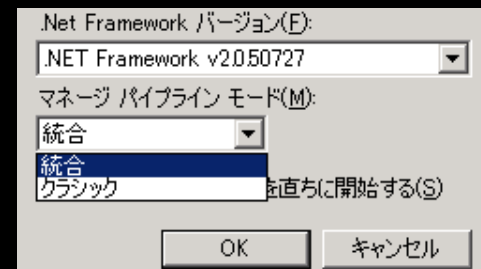
- ▶ マネージモジュールを IIS のパイプラインに組み込むことが可能

Caution!

- ・Context 情報は IIS と ASP.NET パイプラインで統合されていないため、
モジュールの組み合わせによって認証結果を共有できない
→ 統合されたパイプライン処理について熟知する必要がある

IIS 7.0 移行時の設定

- ▶ モードの選択
 - ▶ ワーカープロセス単位で指定可能
- ▶ クラシックモードへの移行
 - ▶ アプリケーションを配置するのみ
- ▶ 統合モードへの移行



①アプリケーションがカスタムモジュール・ハンドラを含む場合

→ WebConfig の編集を行う

<system.web> セクションの <httpModules><httpHandlers> 要素を
<system.webServer> セクションの <modules><handlers> へ移行

→ appCmd を用いて統合モードに移行

②アプリケーションが偽装を構成している場合

→ クラシックモードで実行する(統合モードを諦める)

→ 偽装構成を確認のうえ、警告の解除を行う

<system.webServer> セクションで以下を指定

<validation validatedModeConfiguration="false" />

Windows Server 2008 / Vista 移行

ユーザーアカウント制御 (UAC)

Microsoft®
tech.ed
2008

セキュリティの大幅向上

新機能と互換性の問題

- ▶ 容易にセキュリティレベルを向上可能
 - ▶ BitLocker, 強化された Firewall, IE7 の保護モードなど
- ▶ 非互換の発生が考えられるセキュリティ機能
 - ▶ UAC (User Account Control)
 - ▶ Session 0 の分離 (Services Hardening)
 - ▶ UIPI (User Interface Privilege Isolation)

アプリケーションへの影響が大きい
User Account Control にフォーカスし
その対応方法を紹介する

User Account Control とは?

登場の背景

- ▶ Windows XP, Windows Server 2003 のアカウント制御方式
 - ▶ 実行するタスクに応じてログオンのしなおしが必要
 - ▶ 結果、ユーザーは管理者ユーザーを常時利用



マルウェアにシステムを破壊されるリスク

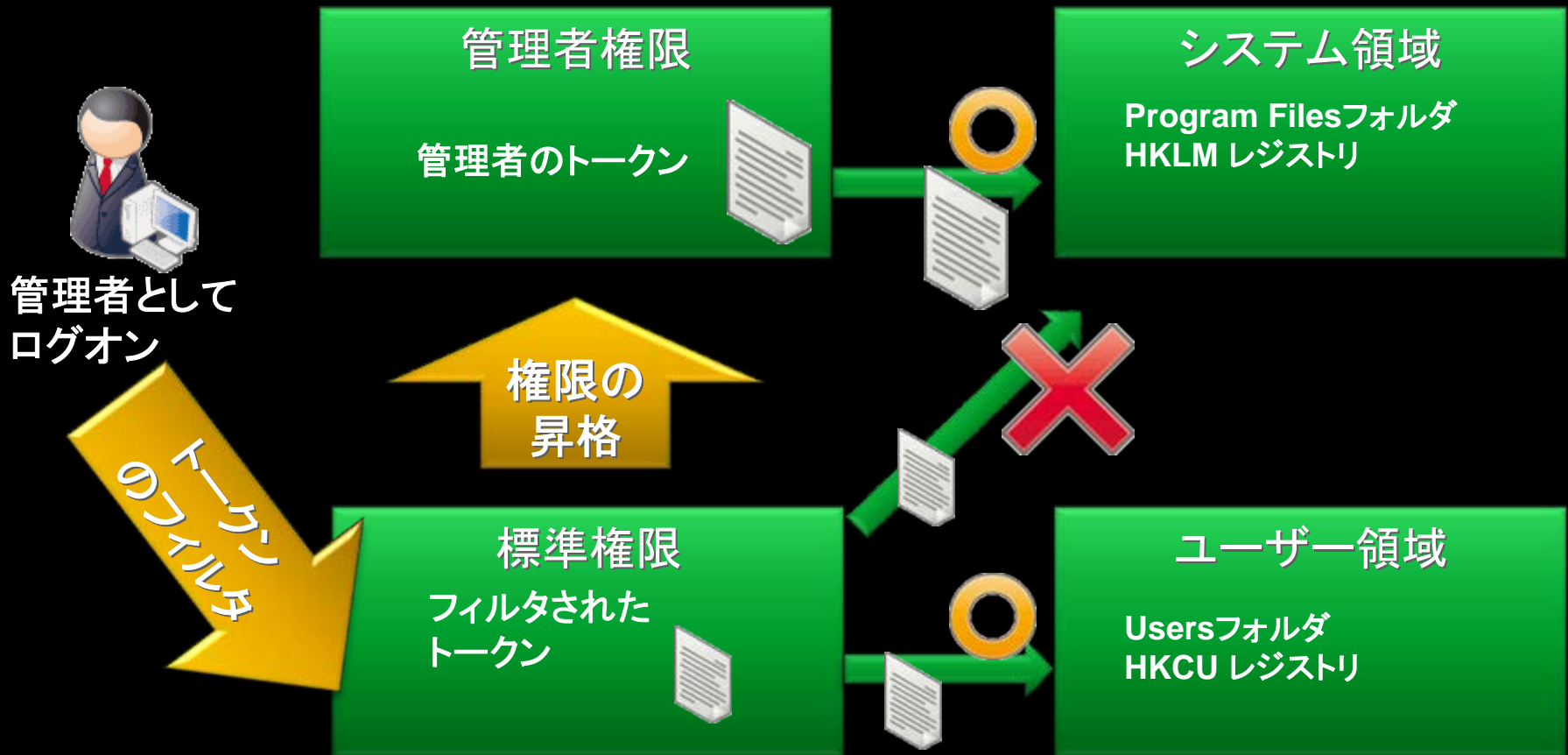
- ▶ Windows Vista, Windows Server 2008 のアカウント制御方式
 - ▶ User Account Control によるアカウント制御
 - ▶ ログオン時の「トークンのフィルタ」
 - ▶ ダイアログによるシームレスな「権限の昇格」



利便性を確保しつつセキュリティの向上を狙う

User Account Control とは?

動作イメージ



管理者権限で動作させることを前提とした
アプリケーションに影響

User Account Control 非互換対応

好ましくない対応方法

①何もしない (互換性保持機能の利用)

- ▶ インストーラーの検知
- ▶ システム領域の仮想化

64bitOS, 次期 OS でサポートされない
動作がわかりにくい

②UAC 無効化 & ビルトイン・アドミニストレーターの利用

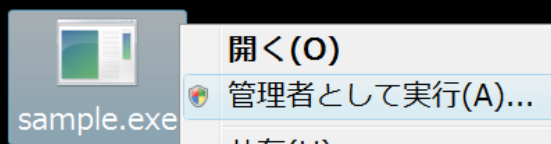
- ▶ Windows Server 2008 では
ビルトイン・アドミニストレーターが既定で有効

Microsoft 非推奨
マシン単位での設定となるため導入済 AP に影響も

User Account Control 非互換対応

好ましい対応方法

- ③「管理者として実行」で昇格する。
 - 実行ファイルを「右クリック」→「管理者として実行」



簡単
ユーザー向けの対応方法

- ④アプリケーション・マニフェストを適用して昇格する。
 - requestedExecutionLevel に requiredAdministrator を指定する
 - 昇格不要の場合でも asInvoker を指定する
 - 互換性保持機能を OFF にするため
 - VS 2005 では mt.exe で、VS 2008 では既定で asInvoker が適用される。



```
<requestedPrivileges xmlns="urn:schemas-microsoft-com:asm.v3">  
<!-- UAC マニフェスト オプション  
Windows のユーザー アカウント制御のレベルを変更するには、  
requestedExecutionLevel ノードを以下のいずれかで置換します。-->  
  
<requestedExecutionLevel level="asInvoker" uiAccess="false" />  
<requestedExecutionLevel level="requireAdministrator" uiAccess="false" />  
<requestedExecutionLevel level="highestAvailable" uiAccess="false" />  
  
<!-- 下位互換性のためにファイルおよびレジストリの仮想化を  
利用する場合は、requestedExecutionLevel ノードを削除してください。-->  
<requestedExecutionLevel level="asInvoker" uiAccess="false" />
```

推奨
開発者向けの対応方法

(参考) UIPI / Session 0 の分離

セキュリティ効果と互換性検証

- ▶ UIPI (User Interface Privilege Isolation)
 - ▶ 高権限で動くアプリケーションに対する
ウインドウメッセージ送信を禁止
 - ▶ 不正なウインドウメッセージによる権限上昇を防ぐ
 - ▶ 整合性レベルと呼ばれる仕組みに基づいて動作する

ウインドウメッセージを利用して
ウインドウ間通信を行っているアプリケーションは検証が必要

- ▶ Session 0 の分離 (Services Hardening)
 - ▶ Windows サービスとユーザーセッションを
Session 0 と Session 1-n に分離
 - ▶ 不正なウインドウメッセージによる権限上昇を防ぐ
 - ▶ Windows サービスからデスクトップに UI を簡単に出せなくなった

ウインドウメッセージもしくはUIを通じて
Windows サービスと対話しているアプリケーションは検証が必要

技術移行

ノータッチデプロイメントから ClickOnce へ

Microsoft®
tech.ed
2008

配置の方法

ノータッチデプロイメントと ClickOnce

▶ ノータッチデプロイメント

- ▶ .NET Framework 1.1 以上で利用可能

▶ ClickOnce

- ▶ .NET Framework 2.0 以上で利用可能

ClickOnce に技術移行することで
様々な機能を取捨選択可能となる

機能比較表

ノータッチデプロイメント vs. ClickOnce

	ノータッチデプロイメント	ClickOnce
配置と更新	○	○
差分のみのダウンロード	△	○
ダウンロードの分散	△	○
オフライン起動	×	○
権限の昇格 (*1)	×	○
HTTP 圧縮	—	○
起動引数の指定	—(*2)	○
ロールバック	—	○
更新時ユーザーデータ引継ぎ	—	△

○: 容易に可能

×: クライアントの設定変更が必要

△: 容易ではないが可能

—: 困難

*1 CAS の制限解除のこと

*2 初期配置は可能だが更新時に問題が起こる

追加機能

.NET Framework と ClickOnce の新機能

- ▶ .NET Framework 2.0 SP1
 - ▶ 公開キートークンが異なる証明書を用いた更新が可能に
- ▶ .NET Framework 3.0
 - ▶ WPF アプリケーションの配布が可能に
 - ▶ XBAP アプリケーションの配布が可能に
- ▶ .NET Framework 3.5
 - ▶ VSTO アプリケーション (Office 2007 用) の配布が可能に

開発環境では Visual Studio 2008 を利用することを推奨

- ▶ Visual Studio 2008 で ClickOnce に機能強化がある
- ▶ クライアント PC には SP を適用することが望ましい (*1)

*1 Visual Studio 2008 インストールにより F/W 2.0 SP1 および F/W 3.0 SP1 が適用されるため

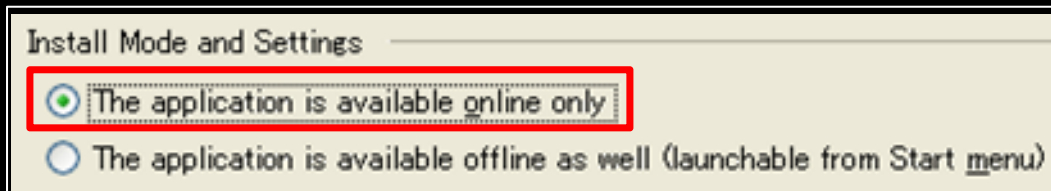
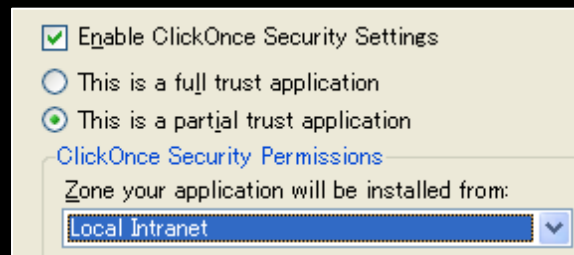
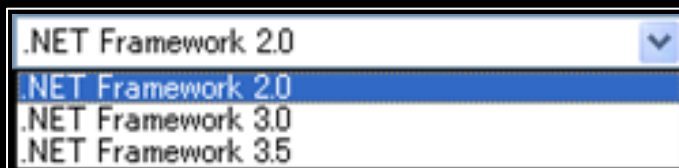
移行方法

ノータッチデプロイメントから ClickOnce へ

▶ ClickOnce への移行

Step 1 : まずは動作を変更せずに移行する

- ▶ .NET Framework 2.0 以上への移行
- ▶ 証明書の調達
- ▶ 権限を昇格させない
- ▶ オンラインモードを指定して発行



移行方法

新たな価値の付与

▶ ClickOnce への移行

Step 2 : ClickOnce の機能を楽しむ

- ▶ ユーザー要求例：通信トラフィックを少なくしたい
 - ▶ 差分ダウンロードを意識した構成および運用手順を策定する
 - ▶ ダウンロードグループを用いて通信を分散する
 - ▶ HTTP 圧縮を利用する
 - ▶ CD-ROM や UNC 共有からの初回インストールを提供する
- ▶ ユーザー要求例：セットアップも ClickOnce で提供してほしい
 - ▶ 必須モジュールを構成する
- ▶ ユーザー要求例：ユーザーごとの設定変更に対応してほしい
 - ▶ データファイルを利用する (*1)
 - ▶ User.config を利用する
- ▶ ユーザー要求例：アプリケーションの更新を強制したい
 - ▶ オンラインモードを利用する
 - ▶ 発行バージョンと必須バージョンを常に同期させる

etc.

*1 Web サーバにデータファイルを配置すると、更新ごとに必ず通信トラフィックが発生する

UAC への対応方針

ClickOnce と UAC

▶ 前提

- ▶ ClickOnce に権限昇格のマニフェストは埋め込めない (*1)
- ▶ ClickOnce はセットアップモジュールを分離できる
- ▶ セットアップ時は UAC が無効となる

▶ 対応方針

- ▶ 管理者権限が必要な処理はセットアップ時に解決する
- ▶ どうしても管理者権限が必要な処理では以下の対応をとる
 - ▶ 管理者権限を要求する処理を別のExeで作成する
 - ▶ 作成した別の Exe にマニフェストを埋め込む
 - ▶ ClickOnce アプリケーションから別プロセスとして呼び出す

ClickOnce に限らず、Vista 対応では
できる限り管理者権限を要求しないことが必要

*1 ClickOnce はユーザー単位でインストールして動作させる。ユーザーには制限ユーザーも含まれるため、管理者権限が必要な動作を ClickOnce に含めることを意識して作られていない。

(マイクロソフト社正式回答)

Windows Server 移行

その他

Microsoft®
tech.ed
2008

MSCS 注意点

- ▶ MSCS クラスタ環境に .NET Framework を導入する場合、クラスタ構築前に必ず .NET Framework を導入しておく
- ▶ 一見、関係のなさそうな KB 934238
 - ▶ 英語版以外の Windows Server 2003 および Windows XP で、.NET Framework 3.0 をインストールした後、[印刷] ダイアログ ボックスの一部の文字が予期せず英語で表示される
<http://support.microsoft.com/kb/934238/ja>

MSCS 注意点

- ▶ KB 934238 問題の箇所
 - ▶ 注：Windows クラスタを使用しているシステムにこの更新プログラムをインストールすると、更新プログラムのインストール時にクラスタ サービスが停止します。
- ▶ MSCS 環境では .NET Framework の言語パックのインストール中にインストーラがエラーで終了する可能性がある

.NET Framework は
クラスタ構築より先にインストールしておく

関連セッション

T2-202 .NET to .NET マイグレーション

～ .NET Framework 1.x から最新環境への移行のために ～

U & U

Users & Unisys

www.unisys.co.jp

UNISYS