Seasar2 を利用した Web アプリケーションの WebOTX AS への 移行ガイド

2008.2.8
第 1.0 版

NEC
第二システムソフトウェア事業部

前書き

【本書の位置付け】

本書は、既存の Web アプリケーションを、NEC 製のアプリケーションサーバである WebOTX に移行する際に参考となる情報を集めたガイドブックです。本書では、Seasar2 を利用した Web アプリケーションのサンプル S2struts-example を例に移行の手順を紹介しています。

移行の際によくある質問に関しては、別資料の「Tomcat から WebOTX AS への移行ガイド」の資料にも説明しています。この資料もご覧ください。 また、移行後のトラブルやチューニングに関しては、別資料の「WebOTX AS Web コンテナ チューニングとトラブルシューティング」の資料もご覧ください。

対象とする移行先は以下のとおりです。

WebOTX V6.x , WebOTX V7.x

改版履歴

版数	年月日	改訂内容	備考
1.0	2008/02/08	第 1.0 版	

目次

1. はじめに

Web システム構築において、コスト削減の面からオープンソース導入が注目されています。しかし、保守運用までを見据えた場合、本当にコスト削減が達成できるのか、という疑問の声もよく耳にします。

WebOTX V6、V7 シリーズでは、サン・マイクロシステムズ社が中心に仕様策定した、エンタープライズ領域の Java 仕様「Java 2 Platform, Enterprise Edition (J2EE) 1.4」に準拠した製品を提供しています。WebOTX は、その機能群の中の Web 層として、Apache Tomcat プロジェクトのオープンソース Tomcat 5.0 をベースにサーブレットや JSP などの Web アプリケーション実行環境(Web コンテナ)を組み込んでいます。

WebOTX V6.3 以降で提供する Web コンテナは、Apache Tomcat 5.0.28 をベースに品質の改良、機能の追加、性能の向上、運用性の強化などを施しています。このため、現在 Apache Tomcat (以下、略して Tomcat と呼びます) 上で動作している Web アプリケーションを少ない労力で WebOTX に移行でき、かつ、システムの信頼性、運用性、性能等の向上を手に入れることができます。

対象読者

本書は、WebOTX 以外のシステム上で動作している既存の Web アプリケーションを、WebOTXへ移行する作業を行う方を対象にしています。

対象とする移行先は以下のとおりです。

WebOTX V6.x, WebOTX V7.x

表記について

パス名表記

本書ではパス名の表記については特に OS を限定しない限りセパレータはスラッシュ' / ' で統一しています。Windows 環境においては' ¥' に置き換えてください。

環境変数表記

インストールディレクトリやドメインルートディレクトリなど環境によって値の異なるものについては環境変数を用いて表します。

${*env*} または $(*env*)で表しています。

例)

${AS_INSTALL} : インストールディレクトリ

${ INSTANCE_ROOT}: ドメインルートディレクトリ

1.1. ソフトウェア条件

WebOTX がサポートするソフトウェアについて説明します。

WebOTXに移行することにより、J2SE SDK やデータベースのバージョンを変更する場合、関連するソフトウェアのバージョンアップの必要性を確認してください。

詳細については、「WebOTX マニュアル　セットアップガイド」ー「1. 使用上の条件」をご覧ください。

1.1.1. 対象プラットフォーム

WebOTX V6.5 がサポートする OS と J2SE SDK のバージョンは次の表のとおりです。

OS	SDK 1.4.2	SDK 5.0
Windows 2000 Server Windows Server 2003	○	○
Windows Server 2003(IPF)	○※	×
Windows Server 2003 x64 Edition	×	○ （Update 4 以降）
HP-UX 11i v1	○※ （1.4.2.05 以降）	○※
HP-UX 11i v2	○ （1.4.2.05 以降）	○ （5.0.02 以降）
Solaris 8 Solaris 9	○※	○※
RHEL AS/ES 4.0	○	○

※WebOTX V6.31 でのサポート

その他のバージョンの対応状況については、WebOTX の Web サイトをご覧ください。

WebOTX V7.11 がサポートする OS と J2SE SDK のバージョンは次の表のとおりです。

OS	SDK 1.4.2	SDK 5.0	Java SE 6
Windows 2000 Server Windows Server 2003	○	○	○
Windows Server 2003(IPF)	後日対応予定	×	×
Windows Server 2003 x64 Edition	×	○ （Update 4 以降）	○
HP-UX 11i v1	○ （1.4.2.05 以降）	○	×
HP-UX 11i v2	○ （1.4.2.05 以降）	○ （5.0.02 以降）	×
Solaris 9 Solaris 10	○	○	○
RHEL AS/ES4.0 MIRACLE LINUX V4.0	○	○	○
RHEL AS/ES4.0（IPF）	○	×	×
RHEL AS/ES4.0（EM64T）	×	○(Update4 以降)	○

1.1.2. Web サーバ

WebOTX がサポートする外部 Web サーバは次の表のとおりです。

Web サーバ	バージョン
WebOTX Web サーバ	1.3.x、2.0.x　※詳細は下表参照
Apache HTTP Server	1.3.x、2.0.x　※詳細は下表参照
Internet Information Server (IIS)	5.0、6.0
Sun Java System Web Server 6.1	6.1
Sun ONE Web Server	6.0 ※HP-UX 11i v2 は未サポート

WebOTX の各バージョンでサポートする WebOTXWeb サーバ、Apache HTTP Server の各バージョンは次のとおりです。

WebOTX	WebOTX Web サーバ、Apache HTTP Server バージョン
WebOTX　V6.1	1.3.31 以降、2.0.52 以降
WebOTX　V6.2	1.3.33 以降、2.0.54　以降
WebOTX　V6.3	1.3.34 以降、2.0.55　以降
WebOTX　V6.4	1.3.36 以降、2.0.58　以降
WebOTX　V7.1	1.3.37 以降、2.0.59 以降

1.1.3. データベース

WebOTX のサポート対象となるデータベースは、次の表のとおりです。また、JDBC データソース、Transaction サービス(JTA)をご使用になる場合、使用するデータベースに対応した JDBC 2.0 または JDBC 3.0 の仕様に準拠している JDBC ドライバをインストールする必要があります。

その他の製品についても、JDBC 2.0 または、JDBC 3.0 の仕様に準拠している JDBC ドライバであれば、使用することができます。

WebOTX V6.5 のサポート対象のデータベース

JDBC ベンダ	JDBC ドライバ・タイプ	サポートするデータベース・サーバ	備考
Oracle	Type 2、4	Oracle8i　R8.1.6（JTA 連携を行う場合は、Oracle R8.1.7 以降）	(*1)
		Oracle R8.1.7	
		Oracle9i Database Release 1 (9.0.1)	(*2)
		Oracle9i Database Release 2 (9.2.0)	
		Oracle Database 10g Release 1 (10.1.0)	
		Oracle Database 10g Release 2 (10.2.0)	
IBM	Type 4	DB2 Universal Database 8.1.4	

Microsoft	Type 4	Microsoft SQL Server 2000	
		Microsoft SQL Server 2005	
Sybase	Type 4	Sybase Adaptive Server Enterprise 12.5	
DataDirect	Type 4	「Connect for JDBC 3.3 以降」経由による Oracle 接続	
	Type 3	「SequeLink for JDBC 5.0」経由による Oracle 接続	
PostgreSQL Development Group	Type 4	PostgreSQL 7.3.2 (JDBC ドライバ 7.3-113) ～ 8.1.2 (JDBC ドライバ 8.1-404) (JTA 連携を行う場合は、バージョン 8.1.0 以降)	(*3)
Cloudscape	Type 4	Cloudscape 3.0.3 (Sun J2EE 1.3.1 SDK にバンドルされるもの)	

(*1) Oracle8i R8.1.6 を使用して 2 フェーズコミットを行う場合には、JDBC データソースのデータソースタイプに「JDBC」を指定して JDBC データソースの JTA 機能を利用してください。 または、JDBC ドライバとして SequeLink を使用してください。

(*2) Oracle の Real Application Cluster(RAC)と X/Open XA の機能を利用して 2 フェーズコミットを行うためには、必ず次のパッチを適用してください。
- PSR 10.1.0.3
- PSR 9.2.0.7

パッチの詳細については Oracle 社の情報をご参照ください。

(*3) PostgreSQL を使用して 2 フェーズコミットを行う場合には、PostgreSQL 8.1 と、バージョン 8.1-404 の JDBC ドライバを使用してください。

WebOTX V7.1 のサポート対象のデータベース

JDBC ベンダ	JDBCドライバ・タイプ	サポートするデータベース・サーバ	備考
Oracle	Type 2、4	Oracle R8.1.7	
		Oracle9i Database Release 1 (9.0.1)	(*1)
		Oracle9i Database Release 2 (9.2.0)	
		Oracle Database 10g Release 1 (10.1.0)	
		Oracle Database 10g Release 2 (10.2.0)	
IBM	Type 4	DB2 Universal Database 8.1.4	
		DB2 V9.1	
Microsoft	Type 4	Microsoft SQL Server 2000	
		Microsoft SQL Server 2005	
Sybase	Type 4	Sybase Adaptive Server Enterprise 12.5	
DataDirect	Type 4	「Connect for JDBC 3.3 以降」経由による Oracle 接続	
	Type 3	「SequeLink for JDBC 5.0」経由による Oracle	

		接続	
PostgreSQL Development Group	Type 4	PostgreSQL 7.3.2 (JDBC ドライバ 7.3-113) ～ 8.1.2 (JDBC ドライバ 8.1-404) (JTA 連携を行う場合は、バージョン 8.1.0 以降)	(*2)
Cloudscape	Type 4	Cloudscape 3.0.3 (Sun J2EE 1.3.1 SDK にバンドルされるもの)	
Apache Derby	Type 4	Apache Derby 10.2.2.0	

(*1) Oracle の Real Application Cluster(RAC)と X/Open XA の機能を利用して 2 フェーズコミットを行うためには、必ず次のパッチを適用してください。
- PSR 10.1.0.3
- PSR 9.2.0.7

パッチの詳細については Oracle 社の情報をご参照ください。

(*2) PostgreSQL を使用して 2 フェーズコミットを行う場合には、PostgreSQL 8.1 と、バージョン 8.1-404 の JDBC ドライバを使用してください。

1.1.4. 同梱するライブラリ

WebOTX には、次のライブラリが同梱されています。

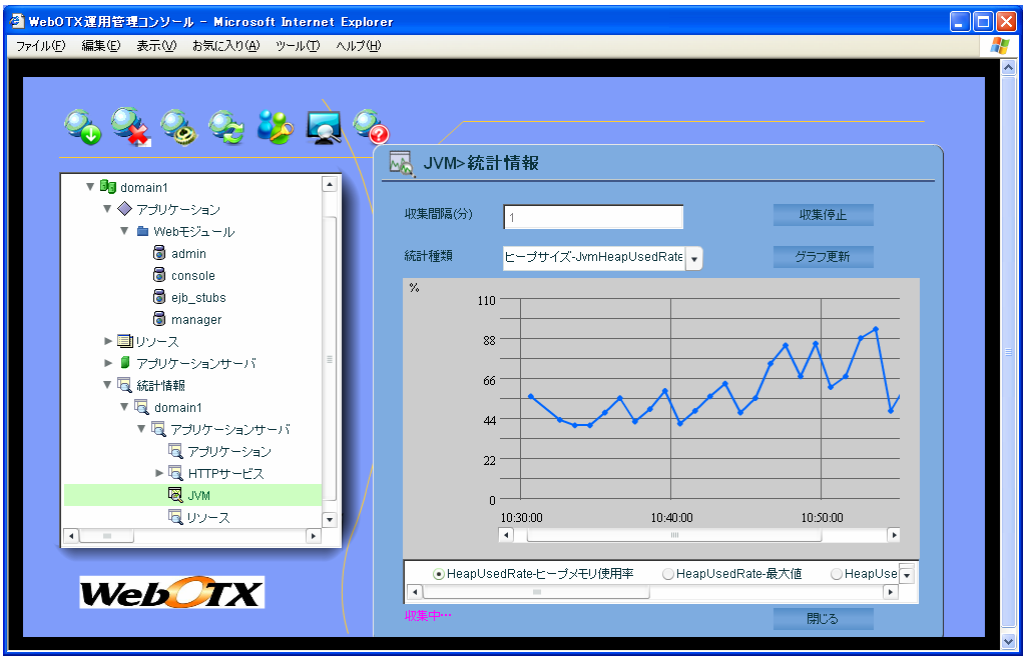
WebOTX に同梱しているライブラリとバージョン

名称	V6.1	V6.22	V6.31	V6.4	V6.5.02	V7.1
Java Beans Activation Framework (JAF)	1.0.2	1.0.2	1.0.2	1.0.2	1.0.2	1.0.2
JavaMail	1.3.1	1.3.1	1.3.1	1.3.1	1.3.1	1.3.1
JAXP	1.2.5	1.2.5	1.2.5	1.2.5	1.2.5	1.3.04
Ant	1.5.4	1.5.4	1.5.4	1.5.4	1.5.4	1.6.5
Jakarta Commons BeanUtiles	1.6.1	1.6.1	1.7.0	1.7.0	1.7.0	1.7.0
Jakarta Commons Codec	1.2	1.3	1.3	1.3	1.3	1.3
Jakarta Commons Collections	2.11	2.11	2.11	2.11	2.11	2.11
Jakarta Commons Digester	1.5	1.6	1.7	1.7	1.8	1.8
Jakarta Commons Discovery	0.2	0.2	0.2	0.2	0.4	0.4
Jakarta Commons EL	1.0	1.0	1.0	1.0	1.0	1.0

Jakarta Commons FileUpload	1.0	1.0	同梱なし	同梱なし	同梱なし	同梱なし
Jakarta Commons Launcher	1.0-dev	1.1	1.1	1.1	1.1	1.1
Jakarta Commons Logging	1.0.4	1.0.4	1.0.4	1.0.4	1.1	1.1
Jakarta Commons Modeler	1.1	1.1	1.1	1.1	2.0	2.0
Jakarta Regexp	1.3	1.3	1.4	1.4	1.4	1.5
Log4j	1.2.8	1.2.8	1.2.13	1.2.13	1.2.14	1.2.14
XML Xerces 2 Java Parser	2.5.0	2.5.0	2.5.0	2.5.0	2.5.0	2.9.0
XML Xalan Java 2	2.5.2	2.5.2	2.5.2	2.5.2	2.5.2	2.7.0
The Web Services Description Language for Java Toolkit (WSDL4J)	1.4	1.4	1.4	1.4	1.4	1.4

1.1.5. Web 版統合運用管理コンソールおよび Flash

WebOTX では Web ブラウザベースの統合運用管理コンソールを提供しています。その Web 版統合運用管理コンソールは Adobe(Macromedia)の Flash ベースで動作します。このため、Web 版統合運用管理コンソールを利用するには、クライアントに Web ブラウザと Macromedia Flash Player 7 以上(Adobe(Macromedia) Flash Player 最新版推奨)が必要です。 Macromedia Flash Player 7 より前のバージョンの Macromedia Flash Player では動作を保証しておりませんのでご注意ください。



Adobe Flash Player は、Adobe社.のWebサイトよりダウンロードしてください。

インターネットに接続できない環境でご利用になる場合は、インターネットに接続できる環境でAdobe Flash Player をダウンロード後、ご利用になるマシンに適用をお願い致します。
Adobe Flash Player のダウンロードは、Adobe社のWebサイトで、「ダウンロード」の「Adobe Flash Player」を選択して「Adobe Flash Player の再配布」へ進み、プラットフォーム等の必要事項を選択して行うことができます。
（このAdobe社のWebサイトの情報は 2007 年 7 月 3 日現在の情報です。）

詳細は、Adobe社のWebサイトを参照してください。

1.1.6. クラスのロードについて

Web アプリケーションで利用する Java のクラスを実行するためにロードを行う機能としてクラスローダがあります。クラスローダには階層があります。上位のクラスローダでロードされたクラスからは下位のクラスローダでロードされたクラスは参照することができません。逆に、下位のクラスローダでロードされたクラスからは、上位のクラスローダでロードされたクラスは参照することができます。

WebOTX のクラスローダの階層は次のようになります。

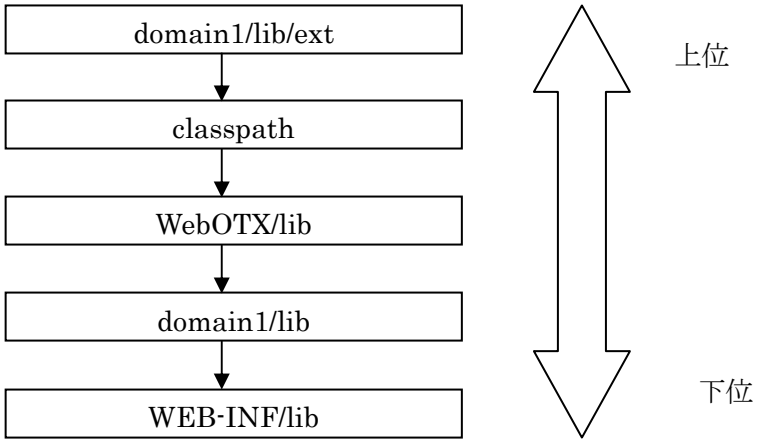


図 1 WebOTX のクラスローダの階層

また、nec-web.xml の delegate の設定により、ロードする優先順位を決定することができます。同じ名前のライブラリがあると、競合が発生し、優先順位の高いパスに含まれるのライブラリが利用されます。ロードされるライブラリの優先順位は delegate=true の場合は以下のようになります。delegate の設定は WebAP の WEB-INF/nec-web.xml で設定します。

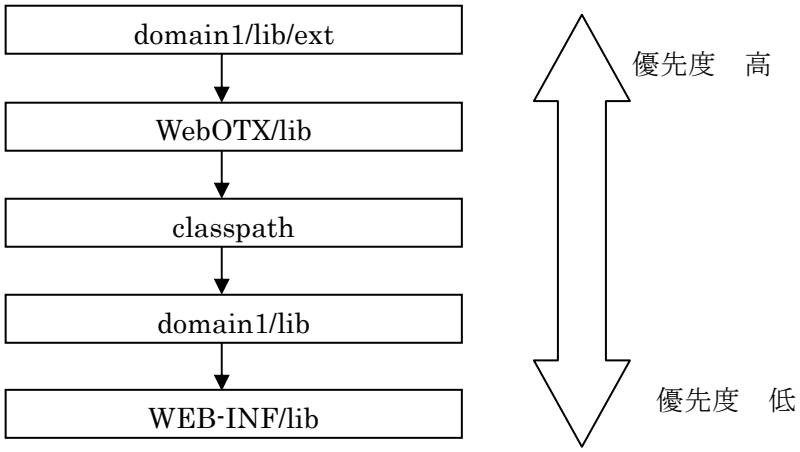


図 2 delegate=true の場合の優先順位

delegate=false の場合は以下のようになります。

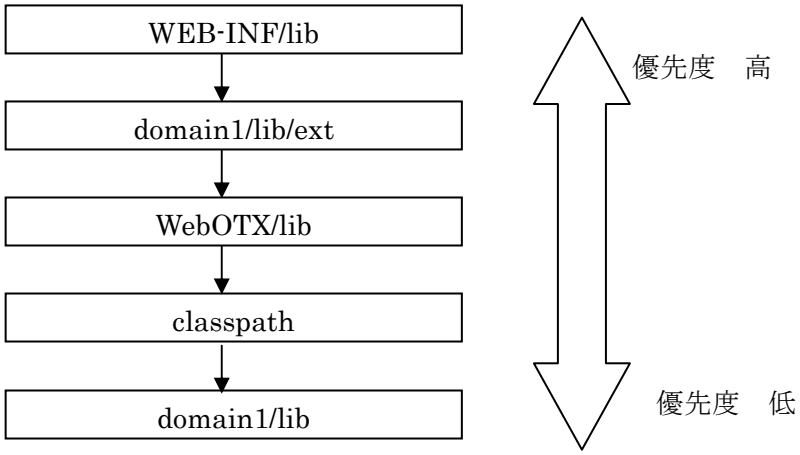


図 3 delegate=false の場合の優先順位

2.移行作業

本章では、既存の Web アプリケーションを WebOTX に移行する際に必要となる作業とその手順について、Seasar2 のサンプルアプリケーション S2struts-example を実例にして説明します。

2.1. 移行作業の概要

既存の Web アプリケーションを他のシステムから WebOTX に移行するための作業と、本書での説明の対応は以下のとおりです。

移行作業	作業の必要性	本書での説明
Web アプリケーションのアーカイブ(WAR)の作成	△ 既に WAR ファイルになっている場合は不要です	2.2.1.章
パッケージの設定	△	2.3 章
JDBC データソースの設定	△	2.4章
log4j の設定	△	2.5章
nec-web.xml の設定	△	2.5 章
セキュリティポリシーの設定	○	2.7章

○:作業要　△:条件により作業要

Web アプリケーションのアーカイブ(WAR)の作成

Web アプリケーションが WAR 形式になっていない場合は、WAR を作成します。

セキュリティポリシーの設定

WebOTX は、インストール・デフォルトの定義により Java セキュリティ・マネージャが動作します。Java セキュリティ・マネージャが働くことで、Web アプリケーションの実行中に、システム上のファイルや他システムへの接続などの資源に対する不用意なアクセスを制限できます。このため、現行システムでセキュリティポリシーを設定していない場合は、WebOTX 移行に際してはセキュリティポリシーの設定が必要となります。 また、現行システムでセキュリティポリシーを設定している場合は、既存のセキュリティポリシー設定を移行する必要があります。

パッケージの設定

既存の JSP コードにおいて、他のクラスをインポート(import)している場合にはパッケージを指定しなければなりません。これは JDK 1.4 から Javac コンパイラが Java 言語仕様へ厳密にチェックするように変わったことに起因します。J2SE 1.4.0 より前のバージョンでは、パッケージの指定は必要ありませんでしたが、J2SE 1.4.0 以降ではパッケージの指定が必要になっています。

JDBC データソースの設定

Tomcat では、「conf/server.xml」ファイルを直接修正することで JDBC データソースの設定を行いますが、WebOTX では、JMX (Java Management Extensions) に対応した統合運用管理ツール/コンソールや統合管理コマンドで設定します。

WebOTX では、データベース・サーバへの接続をファクトリ化し、JNDI API 経由で接続オブジェクト参照を得る仕組みとして JDBC データソースを提供しています。JDBC データソースは、分散トランザクションやコネクション・プーリングなどを考慮しているため、単に JDBC ドライバのインタフェースを呼び出す場合よりも機能拡張されています。

既存の Web アプリケーションにおいて JDBC データソースを使用している場合、JDBC データソースを利用するための設定

を行う必要があります。

log4j の設定

WebOTX は、Apache Logging Server プロジェクトの log4j をバンドルしており、WebOTX 自身のロギング機能に log4j を利用しています。このため、既存の Web アプリケーションで log4j を利用してログ出力している場合、WebOTX の log4j 定義ファイルを設定する必要があります。

nec-web.xml の設定

既存の Web アプリケーションにおいて、Struts や log4j、jakarta commons などのライブラリ JAR ファイルや、WebOTX 以外の CORBA ライブラリを WAR ファイル内の「WEB-INF/lib」配下に格納している場合、また、それらの log4j と CORBA ライブラリについては、WebOTX に含まれるライブラリと重なる場合、Web アプリケーションを配備できないなどの問題が発生することがあります。Web アプリケーションを、そのままのファイル構成でご利用いただくには、nec-web.xml でライブラリのロードに関する設定を行います。

2.2. サンプルアプリケーションによる移行作業の実例

既存の Web アプリケーションとして、Seasar2 のサンプルアプリケーション S2strutsExample V1.2.11 （S2struts-example）を実例にして説明します。

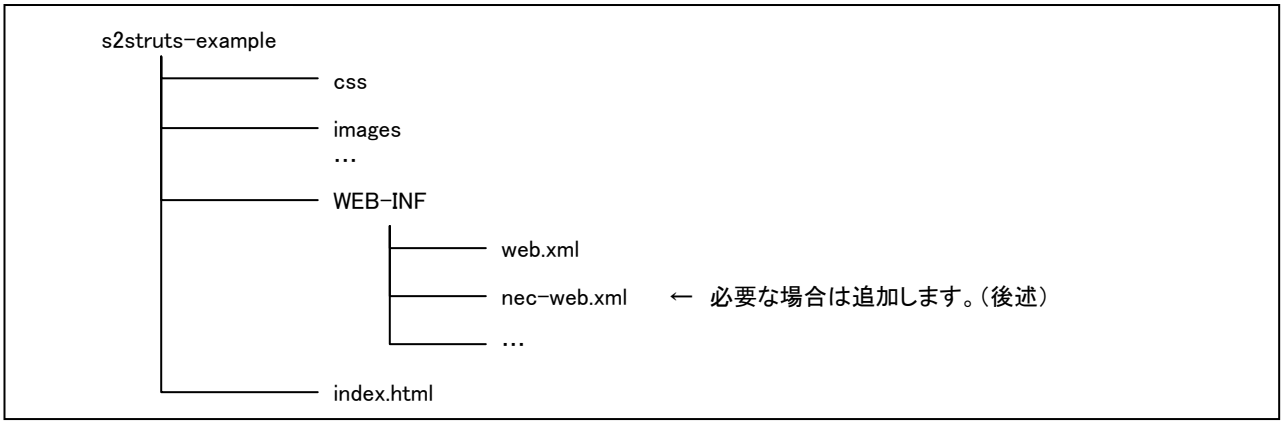
対象のアプリケーションの入手元 ： http://s2struts.seasar.org/ja/index.html

対象のサンプルアプリケーション ： http://s2struts.seasar.org/download/2008-01-16/S2StrutsExample-V1.2.11.zip

2.2.1. Web アプリケーションの準備

対象のサンプルアプリケーションを任意ディレクトリに配置します。 このサンプルの場合は ZIP 形式で配布されていますので、展開します。展開すると下のような、ディレクトリ構成になります。**「動作する Web アプリケーションのアーカイブ（WAR）ファイル」を作成するには、このソースをコンパイルする必要があります。**ここでは Tomcat 上で動作する Web アプリケーションは既にできているものとし、コンパイルの方法については説明しません。 なお、コンパイルには Eclipse と TomcatPlugin が必要です。 Web アプリケーションのアーカイブ（WAR）形式にでない場合は、次のようにjarコマンドで、Web アプリケーションのアーカイブ（WAR）ファイルを作成してください。下の例では、s2struts-example.war を作成します。JDK のインストールディレクトリは”C:¥j2sdk1.4.2”とします。

```
>cd  < s2struts-example をフルパスで指定>        Web アプリケーションを展開したディレクトリに移動します
> C:¥j2sdk1.4.2¥bin¥jar  cvf ../s2struts-example.war ./
```



2.2.2. Web アプリケーションの配備

Web アプリケーション（s2struts-example.war）を次の運用管理コマンド(otxadmin)で配備します。 詳細は、“3.2. 配備方法”を参照してください。

```
otxadmin> deploy --user admin --password adminadmin
              --host localhost --port 6212  s2struts-example.war
```

※配備に失敗する場合は、ログを調査し原因を特定します。

対象のログは、つぎのファイルです。

Web Edition,Standard-J Edition, Stanadard Edition,Enterprise Edition シングルプロセスモード：

${ INSTANCE_ROOT}/logs/webotx_agent.log

Stanadard Edition,Enterprise Edition マルチプロセスモード：

${ INSTANCE_ROOT}/logs/tpsystem/<アプリケーショングループ名>/<プロセスグループ名>/ <プロセスグループ名>.<日時>.<PID>.log

2.2.3. セキュリティポリシーの追加

次のセキュリティポリシに、以下の太字の部分を追加設定します。 設定方法の詳細は、“2.7. セキュリティポリシの設定方法”を参照してください。

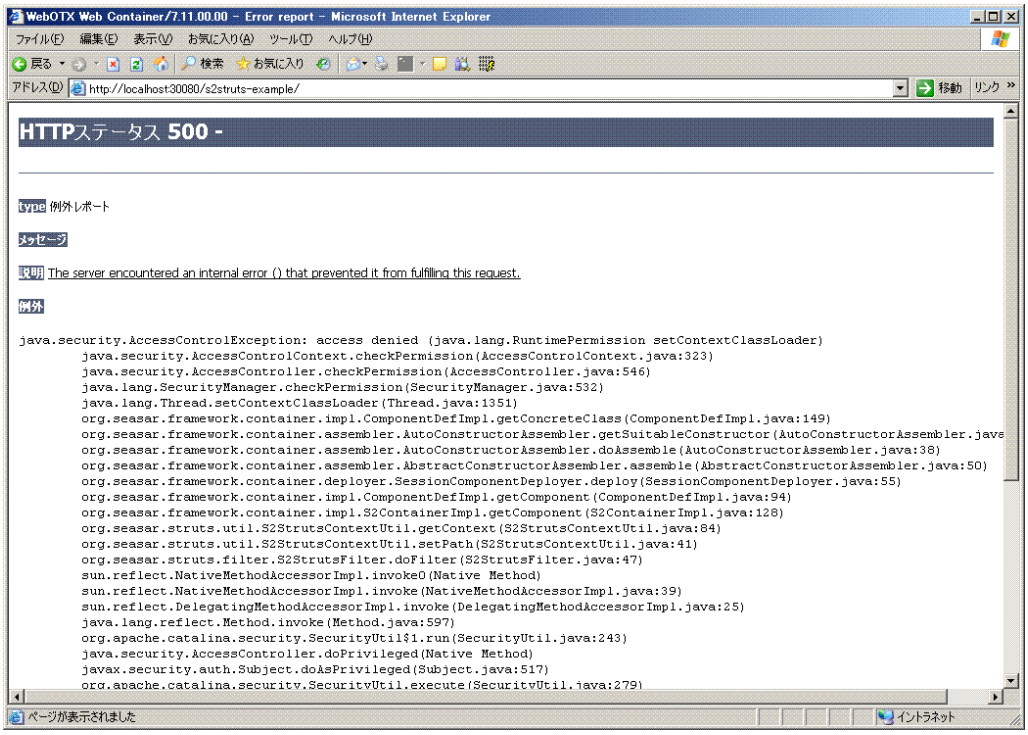
```
grant {
    permission java.lang.RuntimePermission  "loadLibrary.*";
:省略
    permission java.io.FilePermission      "<<ALL FILES>>", "read,write";

// add for S2strats-example
    permission java.lang.RuntimePermission "setContextClassLoader";
    permission java.lang.RuntimePermission "createClassLoader";
    permission java.lang.RuntimePermission "getProtectionDomain";
    permission ognl.OgnlInvokePermission "*";

    permission java.io.FilePermission
"${com.nec.webotx.instanceRoot}${/}lib${/}databases${/}-", "delete";
```

Web アプリケーションにアクセスすると下のように例外が発生することがあります。 画面やログに、このようなアクセス違反を示す情報が出力される場合は、ログの情報に従って必要なアクセス権をセキュリティポリシに追加します。

※ログに出力されない場合は、「2.7.2.セキュリティ例外の調査方法」も、利用して調査してください。

java.security.AccessControlException: access denied (**java.lang.RuntimePermission setContextClassLoader**) と出る場合は、セキュリティポリシに次の行を追加します。

permission **java.lang.RuntimePermission "setContextClassLoader"**;



2.2.4. クラスロードの優先順位

「2.2.5. ライブラリの配置」とあわせて、クラスロードの優先順位を決めます。 上位のクラスローダを優先して読み込む（delegate="true"）か、 下位のクラスローダ（WebAppClassloader）を優先して読み込む（delegate="false"）かを検討します。設定方法については「2.6.追加になった設定ファイル（nec-web.xml）の設定」を参照してください。

※　s2struts-example では、delegate="false"に設定します。 デフォルトで delegate="false"になるので設定変更は不要です。

2.2.5. ライブラリの配置

Web アプリケーションで利用する jdbc ドライバなどのライブラリを必要に応じて、配置します。 jdbc ドライバや、Tomcat で common/lib に配置していたライブラリは、${ INSTANCE_ROOT}/lib/ext の下に配置してください。

なお、クラスローダには上下の参照関係と、読み込みの優先順位がありますので、「1.1.6 クラスロードについて」を参照して、適切な場所にライブラリを配置してください。

※　s2struts-example では、必要なライブラリはありません。

※WebOTX 内で使用しているライブラリが、Web アプリケーション内のライブラリと競合するため起動に失敗する場合があります。この場合は、Web アプリケーション内のライブラリを削除してください。 これまでに競合が発生することが報告されているライブラリは次の通りです。

・commons-logging-x.x.x.jar

・log4j-x.x.x.jar

※クラスローダの読み込み優先順位と、上下の参照関係に起因してライブラリを WEB-INF/lib においたままでは正常に動作しないことがあります。 WEB-INF/lib 以下のライブラリを domain1/lib の下に配置することで回避できたケースがあります。ログの情報を参考に該当のライブラリの配置を検討してください。

2.2.6. サンプルの動作

s2struts-example は、Web アプリケーション内に配置された hsql（DB）を利用します。　まず、

${ INSTANCE_ROOT}/applications/j2ee-modules/s2struts-example/WEB-INF/hsql/bin/runHsqldb.bat を実行して DB を起動します。

※　これで、WebOTX を再起動して、ブラウザから http://localhost/s2struts-example/にアクセスすると 、S2Struts を使用したサンプル（四則演算や Employee Management サンプル(HSQLDB 利用)のデモ）を見ることができます。

2.3. パッケージの設定

JSP で他のクラスをインポート（import）している場合には、パッケージの指定が必要です。

これまで Java2 SDK 1.3.1 以前を使用していた場合は、その Javac コンパイラの構文チェックが緩かったために問題とはなりませんでした。しかし、J2SE 1.4.0 からは構文チェックが厳しくなり、パッケージを持たない JSP コードは、JSP を WebOTX に配備した後に行われる JSP コンパイルでエラーとなります。なお、WebOTX V6 は、J2SE 1.4.2 か 5.0 のみをサポートします。

J2SE 1.3.1 以前のバージョンから移行する場合には、import のパッケージの指定を確認してください。

次のようにパッケージの指定のない import はエラーとなります。

```
<%@ page import="Converter,ConverterHome" %>
```

import するクラスにパッケージの指定を追加して、import の指定をそれにあわせて修正してください。

2.4. JDBC データソースの設定

WebOTX の JDBC データソースを利用するためには、WebOTX 運用環境製品で提供している「統合運用管理ツール」か、標準で備わる「運用管理コマンド（otxadmin）」を利用します。ここでは、運用管理コマンドを利用する設定について説明します。

設定内容の詳細や運用機能の詳細は、WebOTX マニュアルの中の「運用管理コマンドリファレンスマニュアル」－「1. 運用管理エージェント運用管理コマンド」、「運用編（コンフィグレーション）」をご参照ください。

2.4.1. JDBC データソースの設定方法

運用管理コマンドより JDBC データソースを JNDI サーバに登録します。

以下は、WebOTX をインストールした際にデフォルトで作成されるドメイン「domain1」上で Oracle データベース・サーバを利用する JDBC データソースを登録するコマンド例です。

```
otxadmin> create-jdbc-datasource --user admin --password adminadmin
        --port 6212 --host localhost
        --dataSourceType JDBCEX_Oracle
        --jdbcMajorVersion 3 --maxPoolSize 10
        --jdbcUserName scott --jdbcPassword tiger
        --dataSourceName "jdbc:oracle:thin:@hostname:1521:ORCL"
        jdbc/MyOracle
```

※ 実際には 1 行で実行してください。

※ 各引数の詳細は、WebOTX マニュアルの「運用管理コマンドリファレンスマニュアル」－「1. 運用管理エージェント運用管理コマンド」の ″create-jdbc-datasource″ を参照してください。

※ *″jdbc:oracle:thin:@hostname:1521:ORCL″と jdbc/MyOracle の* 部分は環境に合わせて変更してください。

登録が成功すると次のように表示されます。

```
> Command create-jdbc-datasource executed successfully.
```

現在の設定を確認する場合は、次のコマンドを実行します。

```
otxadmin> list-jdbc-datasources --user admin --password adminadmin
        --port 6212
```

2.4.2. クラスパスの設定

以下は、1.1.6 項で紹介した classpath(任意のパスに jar を配置)で指定する方法です。 それ以外の特定のパス(domain/lib/ext)などにライブラリを配置する場合は、クラスパスの設定は不要です。

Oracle データベース・サーバに付属する JDBC ドライバ（ojdbc14.jar）が配置されたディレクトリにクラスパスを設定します。

以下は domain1 ドメインにクラスパスを設定するコマンド例です。

現状のクラスパスを確認します。

```
otxadmin> get --user admin --password adminadmin
            --port 6212 --host localhost
            server.java-config.classpath-suffix
```

※ 実際には 1 行で実行してください。

現状のクラスパスに JDBC ドライバのクラスパスを追加設定します。

```
otxadmin> set --user admin --password adminadmin
            --port 6212 --host localhost
            server.java-config.classpath-suffix="…;/temp/ojdbc14.jar"
```

※ *"/temp/ojdbc14.jar"* 部分は環境に合わせて変更してください。

※ 実際には1行で実行してください。

※ set コマンドは指定の値で上書きしますので、既存の設定値を含めて設定してください。

設定が完了したらドメインを再起動してください。

2.5. log4j の設定

Tomcat 上での Web アプリケーションで log4j を利用して、ログを出力していた場合、WebOTX では、設定を変更しなければならない場合があります。この章では、WebOTX V7.1 と WebOTX V6.x について説明します。

2.5.1. WebOTX V7.1、V6.5、V6.4 での log4j の利用

Web アプリケーションに含まれる log4j を利用する場合と、WebOTX にバンドルされている log4j を利用する場合について説明します。

（1）Web アプリケーションに含まれる log4j を利用する場合

Web アプリケーションが利用する log4j 定義ファイルは、WEB-INF/classes 以下に配置してください。以下の手順で設定をおこなってください。

① WebOTX に設定されている log4j.configration の削除

WebOTX で設定されている log4j.configration 定義を削除します。

Windows 環境では以下のコマンドを実行してください。

```
otxadmin > delete-jvm-options -Dlog4j.configuration=
file¥¥:///${com.nec.webotx.instanceRoot}
${file.separator}config${file.separator }log4j.xml
```

UNIX 環境では以下のコマンドを実行してください。

```
otxadmin > delete-jvm-options -Dlog4j.configuration=
      file¥¥://${com.nec.webotx.instanceRoot}
      ${file.separator}config${file.separator}
      log4j.xml
```

② war ファイルの配備

war ファイルを配備してください。

```
otxadmin > deploy ${WEBAPNAME}
```

③ nec-web.xml の確認

war ファイルを配備すると、nec-web.xml が自動作成されます。nec-web.xml は以下にあります。

```
${INSTANCEROOT}/applications/j2ee-modules/${WEBAPNAME}/WEB-INF/nec-web.xml
```

nec-web.xml で delegate=false となっていることを確認してください。Servlet 2.4 仕様の Web アプリケーションを配備した場合は delegate=true となっているので、true の場合は、nec-web.xml を以下のように変更します。

```
<class-loader delegate=false/>
```

変更後、配備前の Web アプリケーションの WEB-INF 以下に nec-web.xml をコピーし、war ファイルを再作成してください。

④ war ファイルの再配備

「③」で war ファイルを再作成した場合には、再配備してください。

```
otxadmin > deploy -force ${WEBAPNAME}
```

⑤ ドメイン停止

ドメインを停止してください。コマンドは、ドメイン名を domain1 とした場合のコマンドの例です。

```
otxadmin > stop-domain domain1
```

⑥ WebOTX にバンドルしている log4j を削除

WebOTX にバンドルしている log4j を削除してください。

${INSTALLDIR}/lib/log4j.jar を削除してください。

⑦ ドメイン起動

ドメインを起動してください。

```
otxadmin > stop-domain domain1
```

（2）WebOTX にバンドルされている log4j を利用する場合

warファイルに log4j を含めず、${INSTALLDIR}/lib/log4j.jar を利用してログ出力を行うには、Web アプリケーションで利用する定義を、${INSTANCEROOT}/config/log4j.xml に追記してください。

このとき、log4j.xml にデフォルトで定義されている内容は削除しないようにしてください。

詳細な説明については、WebOTX V7.1 では、「WebOTX マニュアル 運用編」－「3.運用と操作」をご覧ください。

※Stanadard Edition,Enterprise Edition マルチプロセスモードで log4j を使用する場合は、つぎのファイルに設定します。

${ INSTANCE_ROOT}/config/tpsystem/logconf/<プロセスグループ名>- <プロセスグループ名>/log4j.xml

2.5.2. WebOTX V6.3 以前での Log4j の利用

Tomcat では、Web アプリケーション毎にログ出力方法を制御する場合は、各 Web アプリケーションの WEB-INF/classes 以下に log4j.properties または log4j.xml という名前で設定ファイルを配置し、設定します。 そのほかに、システム全体でのログ出力方法の設定として、Tomcat3.x ならば TOMCAT_OPTS 、Tomcat4.x 以降ならば CATALINA_OPTS 環境変数へ、log4j.configuration システムプロパティを設定する方法もあります。

WebOTX ではログ出力のために以下の構成をとっています。

- WebOTX のログ出力

  WebOTX では、WebOTX のログを出力用に WebOTX 独自の logging モジュールを利用しています。ドメインのエージェントプロセスが利用するログ定義ファイルは以下の通りです。

  ${INSTANCE_ROOT}/cnofig/log4otx.xml

  また Standard/Enterprise Edition において、プロセスグループの Java プロセスが利用するログ定義ファイルは以下の通りです。

  ```
  ${INSTANCE_ROOT}/config/logconf/<アプリケーショングループ名>/<プロセスグループ名>/log4otx.xml
  ```

- ユーザアプリケーションのログ出力

  WebOTX では、log4j とcommons-logging をバンドルしています。WebOTX にバンドルされているlog4j はバージョン1.2.13 で

す。${INSTALL_ROOT}/lib/log4j.jar にモジュールが格納されています。log4j で利用するログ定義ファイルは以下にあります。

```
${INSTANCE_ROOT}/config/log4j.xml
```

また、commons-logging は、${INSTALL_ROOT}/lib/commons-package.jar に含まれています。

以下は、Tomcat4.x で、システム全体で log4j の設定を行っている場合の移行の例です。

【log4.configuration を使用した設定例 (tomcat4.x の例)】

環境変数への設定(unix の場合)

```
export CATALINA_OPTS="-Dlog4j.configuration=file:///home/foo/log4j.xml"
```

【運用管理コマンドでの設定(WebOTX の例)】

```
otxadmin > create-jvm-options --user admin --password adminadmin --port 6212 --host
           localhost
           "-Dlog4j.configuration=file¥:///${com.nec.webotx.instanceRoot}${file.
           separator}config ${file.separator}<log4j 定義ファイル名>.xml"
```

Web アプリケーションで log4j API を呼び出して、利用する log4j 定義ファイルを読み込む方法についての説明は、ここでは省略します。

WebOTX V6(V6.3 まで)はログ出力に log4j を利用しています。また、多くのユーザアプリケーションにおいても、ログ出力モジュールとして log4j が利用されています。そのため、WebOTX 用のログ定義とユーザアプリケーションのログ定義を同じログ定義ファイルに混同させなければなりませんでした。 しかし、WebOTX V6.4 からは WebOTX ログ出力には WebOTX 独自の logging モジュールを利用しています。よって、ユーザアプリケーションでは、log4j のシステムプロパティにユーザが作成したログ定義ファイルを定義することが可能です。

詳細は、「WebOTX マニュアル 運用編(ロギング)」ー「2.6.アプリケーションが log4j を利用する場合の注意点」をご覧ください。

2.6. 追加になった設定ファイル(nec-web.xml)の設定

Web アプリケーションには、Servlet 仕様にしたがって配備記述子 (Deployment Descriptor) と呼ばれる実行の振る舞いを宣言する XML 形式の設定ファイル (web.xml) を付属させます。WebOTX V6、V7 では、その標準の Servlet 仕様に規定された web.xml 要素を拡張した、nec-web.xml を追加しています。

配備する Web アプリケーション (WAR) ファイル内に nec-web.xml をアーカイブしていない場合は、WebOTX に配備すると、自動的に既定の値を備えた nec-web.xml ファイルが各 Web アプリケーションの WEB-INF ディレクトリ配下に格納されます。ライブラリのロードの設定を変更したい場合や、BASIC/FORM 認証をする際のユーザ名や権限の設定を追加したい場合は、nec-web.xml に該当する情報を設定しなければなりません。

nec-web.xml ファイルの格納場所およびファイルの内容は次のとおりです。

${INSTANCE_ROOT}/applications/j2ee-modules/Web アプリケーション毎のディレクトリ/WEB-INF

【nec-web.xml】

```
<?xml version="1.0" encoding="UTF-8"?>
<nec-web-app xmlns="http://java.sun.com/xml/ns/j2ee">
  <context-root>context_name</context-root>
                                ・・・Web アプリケーションのコンテキスト名
  <class-loader delegate="true"/>
                                ・・・委譲モデル(ロード要求を受けた場合、まずは要求を親の
                                クラスローダに委譲)を使用する場合は、true を指定
</nec-web-app>
```

※ delegate のデフォルト値は、配備する Servlert のバージョンに依存し、Servlet2.3 以前は false、Servlet2.4 では ture になります。

nec-web.xml を変更した場合は、変更した nec-web.xml を、Web アプリケーションの WEB-INF 配下に再配置します。

WEB-INF 下に配置後、次のいずれかの方法で配備を行ってください。

・WAR ファイルを作成して配備

・ディレクトリ指定で配備

配備については、「3.2 配備方法」をご覧ください。

2.6.1. ライブラリのロードをする場合の設定

Servlet 2.2 や 2.3 にしたがった Web アプリケーションを WebOTX に移行する場合には、この nec-web.xml への定義でライブラリのロードに関する指定が必要になる場合があります。このアプリケーションは、Tomcat 3.x、Tomcat 4.x を使用していたケースや、Tomcat 5.x で Servlet 2.3 以前の Web アプリケーションを利用していたケースが該当します。

さらに、Web アプリケーションが次の条件に当てはまる場合にも、nec-web.xml の定義を変更する必要があります。

- Struts、log4j のライブラリ JAR ファイルを WEB-INF/lib に格納して利用している
- Jakarta Commons のライブラリ JAR ファイルを WEB-INF/lib に格納して利用している
- WebOTX 以外の CORBA 通信ライブラリを WEB-INF/lib に格納して利用している

上記の条件に当てはまる場合には、nec-web.xml に宣言されている、<class-loader>の"delegate"属性値を「false」から「true」に変更します。

【変更前の nec-web.xml】

```
<?xml version="1.0" encoding="UTF-8"?>
<nec-web-app xmlns="http://java.sun.com/xml/ns/j2ee">
   <context-root>context_name</context-root>
   <class-loader delegate="false"/>
</nec-web-app>
```

【変更後の nec-web.xml】

```
<?xml version="1.0" encoding="UTF-8"?>
<nec-web-app xmlns="http://java.sun.com/xml/ns/j2ee">
   <context-root>context_name</context-root>
   <class-loader delegate="true"/>
</nec-web-app>
```

2.6.2. BASIC 認証、FORM ベース認証を利用する場合の設定

WebOTX では、BASIC 認証か FORM ベース認証を利用する場合、次の箇所を設定します。

- 使用するレルムにユーザ名、パスワード、権限を登録します。
- 必要に応じて login.conf、domain.xml ファイルに使用するレルムを登録します。
- web.xml ファイルでレルムおよびロールを指定します。
- nec-web.xml で、web.xml で指定したロール名に対応するユーザ名(principal)および権限(group)を指定します。

【web.xml の記述例(抜粋)】

```
<security-constraint>
  <display-name>Server Configuration Security Constraint</display-name>
    <web-resource-collection>
       <web-resource-name>Protected Area</web-resource-name>
       <!-- Define the context-relative URL(s) to be protected -->
       <url-pattern>*.jsp</url-pattern>
       <url-pattern>*.do</url-pattern>
       <url-pattern>*.html</url-pattern>
       <url-pattern>/WebAPManage/*</url-pattern>
    </web-resource-collection>
    <auth-constraint>
      <!-- Anyone with one of the listed roles may access this area -->
       <role-name>stRole</role-name>
```

```
    </auth-constraint>
</security-constraint>

<!-- Login configuration uses form-based authentication -->
<login-config>
    <auth-method>FORM</auth-method>
      <realm-name>file</realm-name>

      <form-login-config>
          <form-login-page>/login.jsp</form-login-page>
           <form-error-page>/error.jsp</form-error-page>
      </form-login-config>
  </login-config>

  <!-- Security roles referenced by this web application -->
  <security-role>
      <description>
           The role that is required to log in to the Administration Application
      </description>
      <role-name>stRole</role-name>
  </security-role>
```

nec-web.xml のロールを指定します

【nec-web.xml の記述例(抜粋)】

```
<context-root>BasicAuthServlet24</context-root>

<security-role-mapping>
  <role-name>stRole</role-name>

  <principal-name>admin3</principal-name>

  <group-name>stGroup</group-name>

</security-role-mapping>
```

web.xml で設定したロール名

ログインを許可するユーザ名を指定します。

ログインを許可するグループ名を指定します
※データベースのロールテーブルに登録するロール名に相当します。

※　ロール名などを nec-web.xml に指定する方法は、「WebOTX 運用(コンフィグレーション)マニュアル」ー「6.3.2 Web アプリケーションの設定」をご覧ください。

2.7. セキュリティポリシーの設定

Tomcat は、オプションなしで起動すると、Java セキュリティ・マネージャを使用しない定義で動作します。セキュリティ・マネージャを使用しない場合、実行時にアクセス制御を行わないため、高速に処理できます。その反面、社会インフラでの利用に求められるサーバ・セキュリティが全く働かない問題があります。

WebOTX では、デフォルトでセキュリティ・マネージャが動作します。これにより、Web アプリケーションが不用意に資源（OS 上のファイルや他システムへの接続など）にアクセスすることを制限することができます。ただし、意図した資源へのアクセスを許可する場合には、Java セキュリティポリシーに追加設定する必要があります。

ここでは、Web アプリケーションが必要とする資源へのアクセスを許可させる方法について説明します。

2.7.1. セキュリティポリシーの設定方法

Web アプリケーションが、任意のディレクトリ/ファイルにアクセスする場合の設定を例に説明します。

Web アプリケーションが、「/Temp」ディレクトリとその配下のファイルの読み込みを行う場合、次のファイルに定義を追加します。

【${INSTANCE_ROOT }/config/server.policy】

```
...
// Basic set of required permissions granted to all remaining code
grant {
      permission java.io.FilePermission "${/}Temp", "read";
      permission java.io.FilePermission "${/}Temp${/}-", "read";
      ...
}
...
```

上記では、「/Temp」ディレクトリへの read の許可と、「/Temp」ディレクトリ配下のファイルの read の許可を追加しています。

設定後は、ドメインを再起動することで反映されます。

JDK のアクセス権とポリシーの構文については以下のサイトを参照してください。(J2SE 1.4 の説明)

http://java.sun.com/j2se/1.4/ja/docs/ja/guide/security/permissions.html

http://java.sun.com/j2se/1.4/ja/docs/ja/guide/security/PolicyFiles.html

2.7.2. セキュリティ例外の調査方法

事前にアクセスするディレクトリ/ファイルなどが分かっている場合には、動作させる前に server.policy に定義を追加できます。しかし、どのようなアクセスがあるか分からない場合には、Web アプリケーションを動作させてみて、セキュリティ例外メッセージから追跡します。

例えば、許可されない資源へのアクセスがあると、${INSTANCE_ROOT}/logs/server.log ログファイルに次のような例外メッセージが記録されます。

```
access denied (java.io.FilePermission /Temp read)
```

この場合には、先に説明した「/Temp」ディレクトリへの read を許可するポリシーの追加が必要になります。

さらに、次のように Java VM のオプションを追加することによって、詳細な情報を確認することができます。

以下は domain1 ドメインにセキュリティ違反の詳細を確認する Java VM オプションを追加するコマンド例です。

```
otxadmin> create-jvm-options --user admin --password adminadmin
        --port 6212 --host localhost
        -Djava.security.debug=access¥¥:failure:
```

※ 実際には 1 行で実行してください。

設定後、ドメインを再起動してください。詳細な情報が logs/server.log に出力されますので、「denied」と記録されているものを確認し、セキュリティポリシーの追加を行ってください。

2.7.3. セキュリティポリシー設定の移行

Tomcat でのセキュリティポリシーの設定は conf/catalina.policy で行います。 WebOTX でのセキュリティポリシーの設定は ${INSTANCE_ROOT }/config/server.policy で行います。 現行のシステムでセキュリティポリシーを設定している場合は、この設定内容を移行します。

【conf/catalina.policy 抜粋 （Tomcat の例）】

```
grant {
    permission java.util.PropertyPermission "java.home", "read";
    :省略
};
```

【${INSTANCE_ROOT }/config/server.policy 抜粋 （WebOTX の例）】

```
grant principal javax.management.remote.JMXPrincipal "admin" {
    :省略
    permission java.io.FilePermission "${java.home}${/}..${/}bin${/}-",
        "read,execute";
    :省略
};
```

3.運用管理方法

Tomcat での運用管理では、Web ブラウザから Administration Tool や Manager アプリケーションを利用し、設定の変更は server.xml を直接編集するのが一般的です。

WebOTX では、JMX(Java Management Extensions)による運用管理機能を提供しています。これにより、Tomcat の server.xml に相当する設定項目をコマンドや GUI により容易に運用管理できるようになっています。

運用管理において、Tomcat と WebOTX は次の違いがあります。

項目	Tomcat	WebOTX	本書での説明
Web サーバとの連携	設定ファイルを直接編集	環境設定ツールを提供	3.1章
配備方法	Manager アプリケーションを使うか、webapps ディレクトリへ WAR ファイルをコピー	運用管理コマンド、運用管理コンソールを提供し、autodeploy ディレクトリへのファイルコピーでの配備にも対応	3.2章
複数 Web コンテナの構成	Tomcat を複数インストール	複数のドメインを作成	3.3章
運用管理コマンド	なし	運用管理コマンドを提供	3.4章
運用管理コンソール	Administration Tool、Manager アプリケーションを提供	運用管理コンソールを提供	3.5章

3.1. Web サーバとの連携

WebOTX では、各 Web サーバとの連携に必要なプラグイン(mod_jk)と、連携のための設定作業を支援する環境設定ツールも提供しています。

WebOTX Web サーバを利用する指定でインストールした場合には、インストール時に連携設定ができます。インストール後に、Web サーバとの連携を再設定する場合は Web サーバとの連携設定が必要です。また、WebOTX Web サーバ以外の他の Web サーバと連携する場合には、インストール時に内蔵 Web サーバを選択して、本章の「Web サーバとの連携」を行ってください。

Web サーバとの連携を設定する場合には、環境設定ツールを利用して行います。複数のドメインが存在する場合、この設定はドメイン毎に行います。

設定の前に対象のドメインと Web サーバを停止してください。

【環境設定ツール】 Windows 版

「スタート」-「プログラム」-「WebOTX」の「環境設定ツール」を起動します。

【環境設定ツール】(シェル) UNIX 版

ログイン名 root でログインします。

```
login: root
```

WebOTX のインストールディレクトリ/bin ディレクトリへ移動します。

```
root> cd /opt/WebOTX/bin
```

./setconf.sh と入力し環境設定ツールを起動します。

```
root> ./setconf.sh
```

上記の環境設定を行った後、ドメインを起動して、次のように運用管理コマンド（otxadmin）で設定します。

```
otxadmin> set --user admin --password adminadmin
          --port 6212 --host localhost
          server.http-service.virtual-server.server.http-listeners="
          ajp-listener-1"
```

環境設定ツール、運用管理コマンドでの設定手順の詳細については、「WebOTX マニュアル セットアップガイド」をご覧ください。

3.2. 配備方法

WAR ファイルを配備するには、運用管理コマンドを利用する方法と運用管理コンソールを利用する方法、autodeploy ディレクトリにコピーする方法があります。

本節では、運用管理コマンドで配備する方法について、説明します。

WebOTX に Web アプリケーションを配備するには、WAR ファイルで配備する方法と、WebOTX が動作するサーバ上の Web アプリケーションのディレクトリを指定して配備する方法があります

3.2.1. WAR ファイルでの配備

WebOTX に Web アプリケーションを配備するには、WAR ファイルで配備する方法と、WebOTX が動作するサーバ上の Web アプリケーションのディレクトリを指定して配備する方法があります。ここでは、WAR ファイルで配備する場合について説明します。

WAR ファイルの作成

Web アプリケーションのディレクトリを”D:¥temp¥sampleAP”とします。JDK のインストールディレクトリは” C:¥j2sdk1.4.2”とします。

Windows では、次のようにコマンドを実行して sampleAP.war ファイルを作成します。

```
cd D:¥temp¥sampleAP
C:¥j2sdk1.4.2¥bin¥jar -cfv sampleAP.war *
```

運用管理コマンドでの配備

運用管理コマンドで WAR ファイルを配備するには次のようにコマンドを実行します。

自ホストでデフォルト作成した domain1 に sampleAP.war を配備する場合の例です。

```
otxadmin> deploy --user admin --password adminadmin
          --host localhost --port 6212  sampleAP.war
```

※ 実際には1行で実行してください。

※ 各引数の詳細は「WebOTX マニュアル 運用管理コマンドリファレンスマニュアル 1.運用管理エージェント運用管理コマンド」をご覧ください。

※Stanadard Edition,Enterprise Edition マルチプロセスモードの場合は、アプリケーショングループ、プロセスグループを指定して配備します。

3.2.2. ディレクトリ指定での配備

運用管理コマンドでディレクトリ指定によって Web アプリケーションを配備するには次のようにコマンドを実行します。

自ホストでデフォルト作成した domain1 に/home/temp/sampleAP ディレクトリ配下に作成したアプリケーションを配備する場合の例です。

```
otxadmin> deploydir --user admin --password adminadmin
                    --host localhost --port 6212
                    /home/temp/sampleAP
```

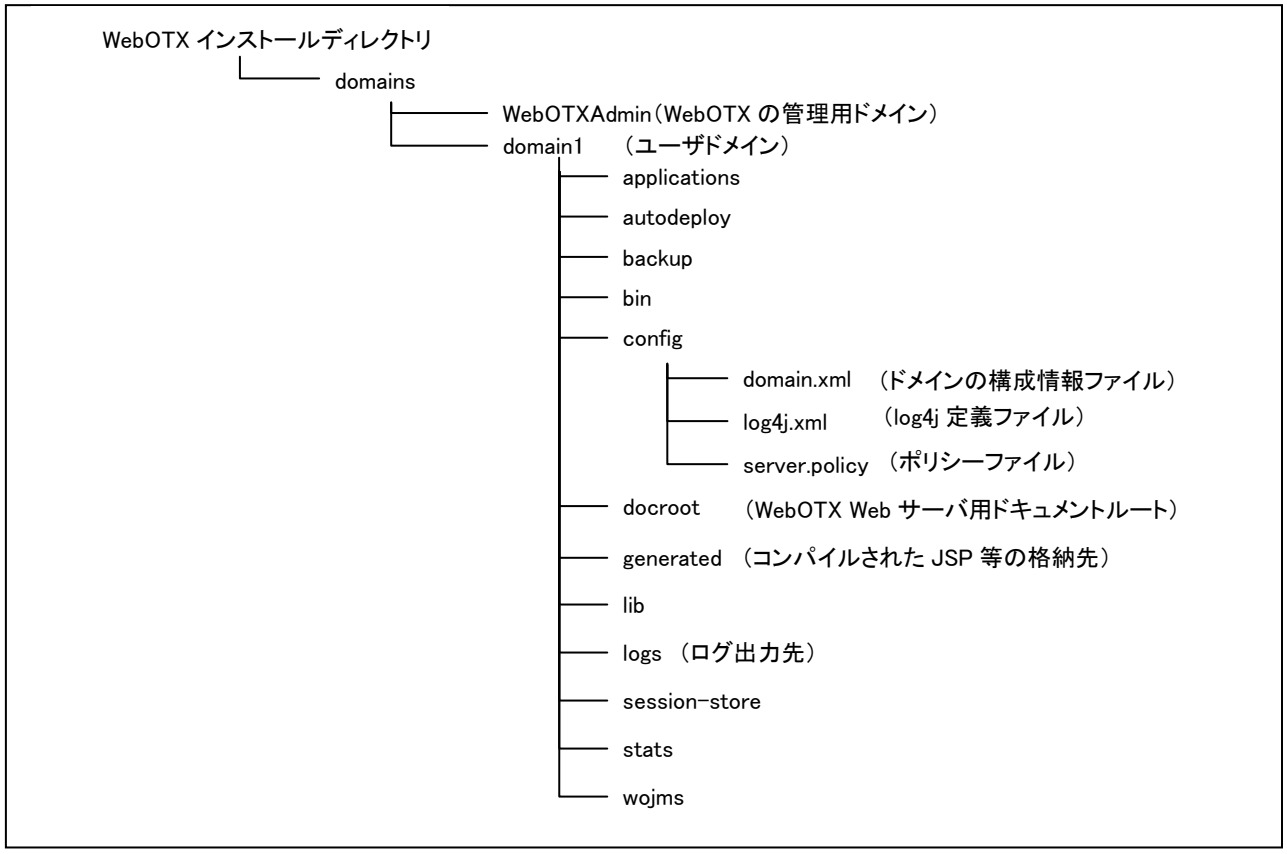
※ 実際には1行で実行してください。

※ 各引数の詳細は「WebOTX マニュアル 運用管理コマンドリファレンスマニュアル 1.運用管理エージェント運用管理コマンド」をご覧ください。

※Stanadard Edition,Enterprise Edition マルチプロセスモードの場合は、アプリケーショングループ、プロセスグループを指定して配備します。

3.3. ディレクトリ構成

WebOTX のドメインに関連する主要なディレクトリは次のようになっています。



3.4. 運用管理コマンド

WebOTX では、運用のための様々なコマンドを用意しています。

domain の起動/停止、構成情報の参照や変更、アプリケーションの配備/配備解除などが運用管理コマンドで可能です。

詳細は「WebOTX マニュアル 運用管理コマンドリファレンスマニュアル」ー「1.運用管理エージェント運用管理コマンド」をご覧ください。

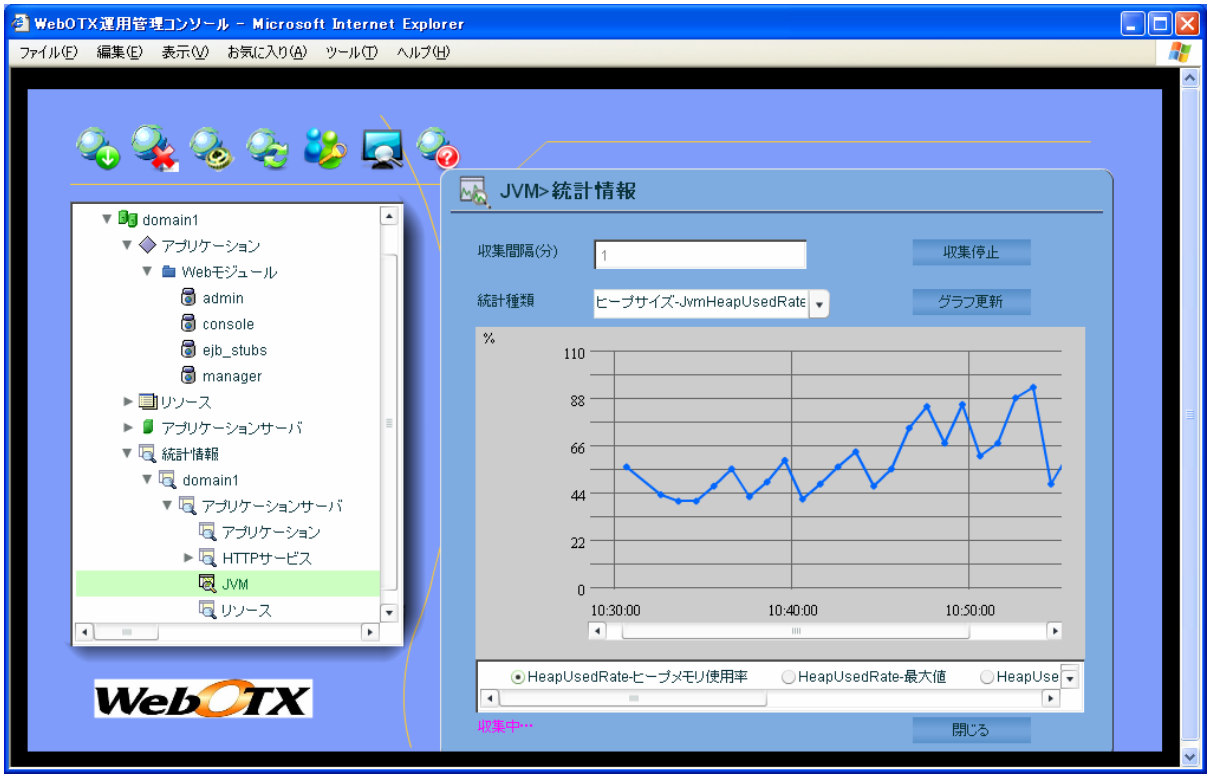
3.5. 運用管理コンソール

Web ブラウザから利用できる Web 版統合運用管理コンソールを用意しています。

詳細は「WebOTX マニュアル WebOTX 運用管理ツールガイド Web 版統合運用管理コンソール」、もしくは Web 版統合運用管理コンソールのヘルプをご覧ください。

Web 版統合運用管理コンソールを利用するには、クライアントに Web ブラウザと Macromedia Flash Player 7 以上(Macromedia Flash Player 最新版推奨)が必要です。

【Web 版統合運用管理コンソール】



4. Q&A

Tomcat からの移行時によく起きる問題や Tomcat からの移行以外にもよくある質問に対する Q&A をまとめました。

4.1. Web アプリケーションの実行エラーに関する Q&A

Q1 Web アプリケーションを実行すると ClassCastException が発生するのですが、どのような原因が考えられますか？

A1 対象となるクラスのロードが正しく行われていない可能性があります。nec-web.xml の設定を変更して、クラスのロード処理を変えて、アプリケーションの動作を確認してください。
設定の変更については、本資料の「2.5.章　追加になった設定ファイル（nec-web.xml）の設定」を参照してください。

Q2 Web アプリケーションを実行するとセキュリティ例外が発生するのですが、どのような原因が考えられますか？

A2 セキュリティポリシーの追加が必要です。本資料の「2.7 セキュリティポリシーの設定」をご覧ください。

Q3 Web アプリケーションからのログが出力されないのですが、どのような原因が考えられますか？

A3 log4jの設定を変更してください。本資料の「2.5 log4jの設定」をご覧ください。

Q4 フィルタがうまく動作しないのですが、どのような原因が考えられますか？

A4 フィルタは Servlet2.3 仕様から追加された仕組みです。web.xml 先頭の DOCTYPE を 2.3 に変更してください。変更後のイメージは次のようになります。

```
<!DOCTYPE web-app
   PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
   "http://java.sun.com/j2ee/dtds/web-app_2_3.dtd">
```

Q5 getParameter で取得したデータが文字化けするのですが、どのような原因が考えられますか？

A5 Tomcat 5.x では、URI のエンコードの指定のために、server.xml で URIEncoding と useBodyEncodingForURI が指定できますが、WebOTX では、server.http-service.http-listener.http-listener-name.property で指定します。
それぞれ、運用管理コマンド（otxadmin コマンド）を利用して、次のように指定します。

```
otxadmin> set server.http-service.http-listener.http-listener-1.property.uri-
          encoding=Windows-31J


otxadmin> set server.http-service.http-listener.http-listener-1.property.use-
          body-encoding-for-uri=true
```

Q6 Web ブラウザで JSP の出力を参照したとき、機種依存文字（㈱など）が文字化けするのですが、どのような原因が考えられますか？

A6 エンコードの指定を Windows-31J にする必要があります。
JSP の場合には、web.xml の<jsp-config>の指定で次のように定義することができます。

【web.xml】

```
<web-app  ....
  <jsp-config>
    <jsp-property-group>
      ....
      <page-encoding>Windows-31J</page-encoding>
    </jsp-property-group>
  </jsp-config>
  ....
  <servlet>
  ....
  </servlet>
</web-app>
```

そのほかに、Servlet での Content-Type による設定、JSP での page ディレクティブの contentType による設定などに起因する文字化けがあります。 詳細については「WebOTX Web コンテナ チューニングとトラブルシューティング」ー「3.1.5. Web アプリケーションで文字化けが発生する」をご覧ください。

Q7 Web アプリケーションを実行すると ClassNotFoundException が発生するのですが、どのような原因が考えられますか？

A7 対象となるクラスが、そのクラスを参照しているクラスから見えなくなっている可能性があります。WebOTXでは、Jakarta Commons のライブラリが”WebOTX/lib”に格納されているため、そのクラスが使用され、WEB-INF/libのクラスが見えなくなっている可能性があります。WebOTXで同梱しているライブラリについては、本資料の「1.1.4同梱するライブラリ」を参照してください。クラスのロードについては、本資料の「1.1.6クラスのロードについて」を参照してください。
nec-web.xmlの”delegate”指定を変更することにより、クラスのロード処理を変えて、アプリケーションの動作を確認してください。もしくは、WEB-INF/libにあるWebOTXで同梱されているライブラリを削除して、アプリケーションの動作を確認してください。
設定の変更については、本資料の「2.5.章 追加になった設定ファイル（nec-web.xml）の設定」を参照してください。

4.2. Web アプリケーションの開発に関する Q&A

Q1 servlet をコンパイルするときに、WebOTX ではどの jar を利用するのでしょうか？

A1 ${AS_INSTALL}/lib/j2ee.jar を利用してください。

Q2 Web アプリケーションを自動的に配備するには、どうすれば良いのでしょうか？

A2 Tomcat では、webapps ディレクトリに WAR ファイルを置くと自動的に配備されますが、WebOTX では
${ INSTANCE_ROOT}/autodeploy ディレクトリに WAR ファイルを格納することで、自動的に配備できます。

4.3. 環境設定、チューニングに関する Q&A

Q1 WebOTX で JavaVM のメモリ量を指定するには、どうすれば良いのでしょうか？

A1 Tomcat では、起動用のファイルで JavaVM のメモリ量を指定しますが、WebOTX では運用管理コマンド（otxadmin） で指定します。

【割り当てメモリの最大値を 640 MB にする例】

```
otxadmin> create-jvm-options --user <ユーザ名> --password <パスワード> --host <ホスト名> --port <管理ポート> -Xmx640m:
```

詳細については、「WebOTX Web コンテナ チューニングとトラブルシューティング WebOTX V6 編」－「2.1. Java VM への割り当てメモリ」をご覧ください。

Q2 Apache と連携するときの同時接続（スレッド）数の設定は、どうすれば良いのでしょうか？

A2 Tomcat では server.xml を編集して設定しますが、WebOTX では運用管理コマンド（otxadmin） で指定します。

【同時に処理できるリクエストの数を拡大する例】

```
otxadmin> set --user <ユーザ名> --password <パスワード> --host <ホスト名> --port <管理ポート> server.http-service.http-listener.<リスナ ID>.max-processors=<最大数>
```

詳細については、「WebOTX Web コンテナ チューニングとトラブルシューティング WebOTX V6 編」－「2.2. プロセッサ数」をご覧ください。

Q3 JDBC データソースの設定は、どうすれば良いのでしょうか？

A3 Tomcatでは server.xml を編集して設定しますが、WebOTXでの設定方法は本資料の「2.4 JDBCデータソースの設定」をご覧ください。

Q4 1台のサーバで複数の Web コンテナを起動するには、どうすれば良いのでしょうか？

A4 Tomcat では、Tomcat のディレクトリをコピーして、同じマシンで複数の Tomcat を起動しますが、WebOTX では複数のドメインを作成することで複数の Web コンテナを起動することができます。

5. 注意事項

5.1. J2SE SDK に関する注意事項

WebOTX が動作保証する J2SE SDK（JDK）に変更する場合、Web アプリケーションや Web アプリケーションが依存するライブラリが、特定の J2SE SDK に依存しないか動作確認を行ってください。

5.2. Tomcat との差異

- WebOTX では WAR ファイル名から拡張子を除いた名称と、配備時に指定したコンテキスト名が異なった場合、WAR ファイル名から拡張子を除いた名称をコンテキスト名としては利用できません。
- WebOTX では、web.xml ファイルから Web アプリケーションの配備情報を取得しているため、web.xml ファイルは必須です。web.xml ファイルがない、もしくは、正しく記述されていない場合、WAR ファイルを配備することができません。
- クラスローダがTomcatとは異なっています。詳細は、本資料の「1.1.6クラスのロードについて」を参照ください。

信頼性、柔軟性、サポート
3つの安心でお客様のシステムを支えます。

をどうぞよろしくお願いします。

■ お問合わせ先
NEC 第二システムソフトウェア事業部
mailto:info-webotx@isd.jp.nec.com
■ 製品ホームページURL
http://www.nec.co.jp/WebOTX/