3. コンピュータシステムやアーキテクチャに関する知識II

1. 科目の概要

OSS が動作する基盤となるコンピュータシステムとアーキテクチャに関して、代表的なアーキテクチャである Web システムや、OSS を活用したシステム基盤設計の手順や事例を紹介し、さらに将来におけるコンピュータシステムの発展についても説明する。

2. 習得ポイント

本科目の学習により習得することが期待されるポイントは以下の通り。

習得ポイント	説　明	シラバスの対応コマ
II-3-1. 様々な形態のコンピュータシステム	OSSが動作するコンピュータシステムについて、集中システム、分散システムといったコンピュータシステムにおける形態の違いについて解説する。さらに分散システムについては代表的な分散システムであるクライアントサーバシステムとWebシステムについて触れる。	9
II-3-2. 非機能要件とシステム構成	コンピュータシステムを構成する非機能要件について述べる。システム構成で要点となるクラスタ構成や、ネットワーク負荷分散、サーバ負荷分散といった負荷分散構成、データ分散構成によるセキュリティ構成などについて説明する。	9
II-3-3. OSSによるシステムアーキテクチャの実現	代表的なコンピュータシステムの例として、Webサーバ、アプリケーションサーバ、データベースサーバなどがある。これらについて、どのようなOSSを用いて構成すると効果的に実現できるかを、具体的な例を用いて示す。	9
II-3-4. Webシステムのアーキテクチャ	Webシステムの事例を対象として、システムアーキテクチャ検討におけるOSS活用のポイントについて解説する。システムアーキテクチャ導入時のシナリオにおいて検討すべき項目や、実際にOSSがどのように活用されてシステム基盤を構築するかについて述べる。	10
II-3-5. OSSによるWeb3層アプリケーション	Web3層アプリケーションとは何か解説し、Webアプリケーション開発の特徴や利点について説明する。そのうえで、OSS を活用した3層アプリケーションの具体的な構築例を示し、Web3層アプリケーション開発におけるOSSの効果的な利用方法を解説する。	11
II-3-6. OSSによるシステム基盤設計の手順	OSSを活用してシステム基盤を構築する手順を理解させる。OSSプロダクトの選定方法からシステムを構成するソフトウェアの配置、ハードウェア性能の検討、ライセンス検証、非機能要件検討といった具体的な手順を概説する。	12, 13
II-3-7. OSSによるシステム基盤設計事例	「OSSによるシステム基盤設計の手順」において示した手順を、具体的な事例として示す。ビジネスロジックの定義や非機能要件の検証方法など、実際にシステム基盤を設計する際に重要な部分について解説する。	12, 13
II-3-8. OSSが動作するハードウェア	OSSが動作する基盤となるハードウェアについて、サーバ、クライアントといったそれぞれ利用環境に合わせたハードウェアの特徴を整理する。さらに、OSSを動作させるにあたって留意すべきポイントや課題についても述べる。	14
II-3-9. コンピュータ以外への発展	当初は一般のコンピュータがOSSの主たる適用対象であったが、現在では携帯電話やPDA、情報家電といった組み込み分野など様々なアーキテクチャへと対象範囲が拡大している。これらに関して、アプリケーション基盤の構成要素や設計のポイントなどについて説明する。	15
II-3-10. 次世代ネットワークアーキテクチャ	OSSと親和性の高いネットワーク技術も、OSSがその基盤を支えているインターネットから、インターネット以外のネットワークへとOSSの適用範囲が広がっている。ここでは、ユビキタスネットワーク、センサネットワークやIPv6の動向について述べる。	15

【学習ガイダンスの使い方】

1. 「習得ポイント」により、当該科目で習得することが期待される概念・知識の全体像を把握する。
2. 「シラバス」、「IT 知識体系との対応関係」、「OSS モデルカリキュラム固有知識」をもとに、必要に応じて、従来の IT 教育プログラム等との相違を把握した上で、具体的な講義計画を考案する。
3. 習得ポイント毎の「学習の要点」と「解説」を参考にして、講義で使用する教材等を準備する。

3. IT 知識体系との対応関係

「3. コンピュータシステムやアーキテクチャに関する知識Ⅱ」と IT 知識体系との対応関係は以下の通り。

科目名	基本レベル（Ⅰ）								応用レベル（Ⅱ）						
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
3. コンピュータシステムやアーキテクチャに関するスキル	<コンピュータアーキテクチャの基本>	<コンピュータハードウェアの基本>	<CPUアーキテクチャの基本>	<ディスクと周辺機器の基本>	<インタフェース技術の基本>	<ソフトウェアアーキテクチャ>	<OSのアーキテクチャ>	<ミドルウェアの種類と特徴>	<コンピュータシステムの構成>	<システムアーキテクチャの活用事例>	<Webシステムのアーキテクチャ>	<OSSを活用した基盤設計ケースワーク>	<オープンソースシステムアーキテクチャ構築>	<OSSの動作環境としてのハードウェア>	<これからのオープンソースアーキテクチャの動向>

[シラバス：http://www.ipa.go.jp/software/open/ossc/download/Model_Curriculum_05_03.pdf]

<IT 知識体系上の関連部分>

分野		科目名	1	2	3	4	5	6	7	8	9	10	11	12	13
組織関連事項と情報システム	1	IT-IAS 情報保証と情報セキュリティ	IT-IAS1. 基礎的な問題	IT-IAS2. 情報セキュリティの仕組み（対策）	IT-IAS3. 運用上の問題	IT-IAS4. ポリシー	IT-IAS5. 攻撃	IT-IAS6. 情報セキュリティ分野	IT-IAS7. フォレンジック（情報証拠）	IT-IAS8. 情報の状態	IT-IAS9. 情報セキュリティサービス	IT-IAS10. 脅威分析モデル	IT-IAS11. 脆弱性		
	2	IT-SP 社会的な観点とプロフェッショナルとしての課題	IT-SP1. プロフェッショナルとしてのコミュニケーション	IT-SP2. コンピュータの歴史 [3-6]	IT-SP3. コンピュータを取り巻く社会環境	IT-SP4. チームワーク	IT-SP5. 知的財産権	IT-SP6. コンピュータの法的問題	IT-SP7. 組織の中のIT	IT-SP8. プロフェッショナルとしての倫理的な問題と責任	IT-SP9. プライバシーと個人の自由				
応用技術	3	IT-IM 情報管理	IT-IM1. 情報管理の概念と基礎 [3-8]	IT-IM2. データベース問合わせ言語	IT-IM3. データアーキテクチャ	IT-IM4. データモデリングとデータベース設計	IT-IM5. データと情報の管理	IT-IM6. データベースの応用分野							
	4	IT-WS Webシステムとその技術	IT-WS1. Web技術	IT-WS2. 情報アーキテクチャ	IT-WS3. デジタルメディア	IT-WS4. Web開発 [3-8]	IT-WS5. 脆弱性	IT-WS6. ソーシャルソフトウェア							
ソフトウェアの方法と技術	5	IT-PF プログラミング基礎	IT-PF1. 基本データ構造	IT-PF2. プログラミングの基本的構成要素	IT-PF3. オブジェクト指向プログラミング	IT-PF4. アルゴリズムと問題解決	IT-PF5. イベント駆動プログラミング	IT-PF6. 再帰							
	6	IT-IPT 技術を統合するためのプログラミング	IT-IPT1. システム間通信	IT-IPT2. データ割り当てと交換	IT-IPT3. 統合的コーディング	IT-IPT4. スクリプティング手法	IT-IPT5. ソフトウェアセキュリティの実現	IT-IPT6. 種々の問題	IT-IPT7. プログラミング言語の概要						
	7	CE-SWE ソフトウェア工学	CE-SWE0. 歴史と概要	CE-SWE1. ソフトウェアプロセス	CE-SWE2. ソフトウェアの要求と仕様	CE-SWE3. ソフトウェアの設計	CE-SWE4. ソフトウェアのテストと検証	CE-SWE5. ソフトウェアの保守	CE-SWE6. ソフトウェア開発・保守ツールと環境	CE-SWE7. ソフトウェアプロジェクト管理	CE-SWE8. 言語翻訳 [3-6]	CE-SWE9. ソフトウェアのフォールトトレランス	CE-SWE10. ソフトウェアの構成管理	CE-SWE11. ソフトエアの標準化	
	8	IT-SIA システムインテグレーションとアーキテクチャ	IT-SIA1. 要求仕様	IT-SIA2. 調達/手配	IT-SIA3. インテグレーション [3-6]	IT-SIA4. プロジェクト管理	IT-SIA5. テストと品質保証	IT-SIA6. 組織の特性	IT-SIA7. アーキテクチャ [3-9]						
システム基盤	9	IT-NET ネットワーク	IT-NET1. ネットワークの基礎 [3-1]	IT-NET2. ルーティングとスイッチング	IT-NET3. 物理層	IT-NET4. セキュリティ	IT-NET5. アプリケーション分野 [3-8]	IT-NET6. ネットワーク管理							
	10	CE-NWK テレコミュニケーション	CE-NWK0. 歴史と概要 [3-1]	CE-NWK1. 通信ネットワークのアーキテクチャ [3-1]	CE-NWK2. 通信ネットワークのプロトコル	CE-NWK3. LANとWAN	CE-NWK4. クライアントサーバコンピューティング	CE-NWK5. データのセキュリティと整合性	CE-NWK6. ワイヤレスコンピューティングとモバイルコンピューティング	CE-NWK7. データ通信	CE-NWK8. 組込み機器向けネットワーク	CE-NWK9. 通信技術とネットワーク概要	CE-NWK10. 性能評価	CE-NWK11. ネットワーク管理	CE-NWK12. 圧縮と伸張
			CE-NWK13. クラスタシステム	CE-NWK14. インターネットアプリケーション	CE-NWK15. 次世代インターネット	CE-NWK16. 放送									
	11	IT-PT プラットフォーム技術	IT-PT1. オペレーティングシステム [3-7]	IT-PT2. アーキテクチャと機構 [3-5]	IT-PT3. コンピュータインフラストラクチャ	IT-PT4. デプロイメントソフトウェア	IT-PT5. ファームウェア	IT-PT6. ハードウェア							
	12	CE-OPS オペレーティングシステム	CE-OPS0. 歴史と概要 [3-7]	CE-OPS1. 並行性	CE-OPS2. スケジューリングとディスパッチ [3-7]	CE-OPS3. メモリ管理 [3-7]	CE-OPS4. セキュリティと保護	CE-OPS5. ファイル管理	CE-OPS6. リアルタイムOS	CE-OPS7. OSの概要	CE-OPS8. 設計の原則	CE-OPS9. デバイス管理	CE-OPS10. システム性能評価		
コンピュータアーキテクチャ	13	CE-CA0 コンピュータのアーキテクチャと構成	CE-CA00. 歴史と概要 [3-1,2]	CE-CA01. コンピュータアーキテクチャの基礎 [3-1,2,4]	CE-CA02. メモリシステムの構成とアーキテクチャ	CE-CA03. インタフェースと通信 [3-3,5]	CE-CA04. デバイスサブシステム	CE-CA05. CPUアーキテクチャ [3-3]	CE-CA06. 性能・コスト評価	CE-CA07. 分散・並列処理 [3-9]	CE-CA08. コンピュータによる計算	CE-CA09. 性能向上			
複数領域にまたがるもの	14	IT-ITF IT基礎	IT-ITF1. ITの一般的なテーマ	IT-ITF2. 組織の問題	IT-ITF3. ITの歴史	IT-ITF4. IT分野（学科）とそれに関連のある分野（学科）	IT-ITF5. 応用領域	IT-ITF6. IT分野における数学と統計学の活用							
	15	CE-ESY 組込みシステム	CE-ESY0. 歴史と概要	CE-ESY1. 低電力コンピューティング	CE-ESY2. 高信頼性システムの設計	CE-ESY3. 組込み用アーキテクチャ	CE-ESY4. 開発環境	CE-ESY5. ライフサイクル	CE-ESY6. 要件分析	CE-ESY7. 仕様定義	CE-ESY8. 構造設計	CE-ESY9. テスト	CE-ESY10. プロジェクト管理	CE-ESY11. 並行設計（ハードウェア、ソフトウェア	CE-ESY12. 実装
			CE-ESY13. リアルタイムシステム設計	CE-ESY14. 組込みマイクロコントローラ	CE-ESY15. 組込みプログラム	CE-ESY16. 設計手法	CE-ESY17. ツールによるサポート	CE-ESY18. ネットワーク型組込みシステム	CE-ESY19. インタフェースシステムと混合信号システム	CE-ESY20. センサ技術	CE-ESY21. デバイスドライバ	CE-ESY22. メンテナンス	CE-ESY23. 専門システム	CE-ESY24. 信頼性とフォールトトレランス	

4. OSS モデルカリキュラム固有の知識

OSS モデルカリキュラム固有の知識として、具体的にオープンソースを用いたシステム構築事例やその費用対効果などの検討が挙げられる。ハードウェアとオープンソースの関わりについても固有の知識となる。

科目名	第9回	第10回	第11回	第12回	第13回	第14回	第15回
3.コンピュータシステムやアーキテクチャに関する知識Ⅱ	(1)コンピュータシステムの構成 (2)非機能要件からの構成 (3)オープンソース化の構成の例	(1)システムアーキテクチャ導入のシナリ (2)OSS を用いたシステム基盤事例	1)Web アプリケーション開発の特長と利 (2)インターネット技術の効果的な導入方法	(1)OSS プロダクトの選定 (2)プロダクトの特性を活かしたソフトウェア (3)ハードウェア構成とそのメリット (4)コスト面でのメリット (5)ライセンスの検証 (6)非機能要件の検討	(1)ハードウェア構成とそのメリット (2)OSS プロダクトの導入 (3)アプリケーションの配置 (4)非機能要件の検証	(1)オープンソースハードウェア (2)ハードウェアインタフェースのライセンス	(1)非コンピュータによる基盤 (2)非インターネット

（網掛け部分は IT 知識体系で学習できる知識を示し、それ以外は OSS モデルカリキュラム固有の知識を示している）

スキル区分	OSS モデルカリキュラムの科目	レベル
基礎分野	3　コンピュータシステムやアーキテクチャに関する知識 II	応用
習得ポイント	II -3-1. 様々な形態のコンピュータシステム	
対応する コースウェア	第 9 回 (コンピュータシステムの構成)	

II -3-1. 様々な形態のコンピュータシステム

OSS が動作するコンピュータシステムについて、集中システム、分散システムといったコンピュータシステムにおける形態の違いについて解説する。さらに分散システムについては代表的な分散システムであるクライアントサーバシステムと Web システムについて触れる。

【学習の要点】

* コンピュータシステムのアーキテクチャは集中システムと分散システムに分類することができる。
* 集中システムとは、メインフレームと呼ばれる大型コンピュータでデータの処理を集中的に行うシステムである。
* 分散システムとは、ネットワークを通じて複数のコンピュータが協調して処理を行うシステムである。
* クライアントサーバシステムは分散システムの一種であり、特定の処理を集中的に担当するサーバソフトウェアとユーザが使用するコンピュータ上のクライアントソフトウェアが協調して処理を行うシステムである。
* Web システムはクライアントサーバシステムの構成要素の内、クライアントソフトウェアとして Web ブラウザを用いるシステムである。

	集中システム	クライアント サーバシステム	Webシステム
コスト	×	○	○
処理 方法	大型コンピュータによる集中処理	クライアントとサーバの分担処理	サーバによるビジネスロジック処理 & ブラウザによる表示処理
運用 管理	◎ 高度な運用管理、セキュリティ管理の仕組みが利用可能	× 複数コンピュータを管理する仕組みを用意する必要があり、煩雑	△ クライアントの管理は容易だが、サーバの管理は煩雑
柔軟性	△ 構成変更に伴うコストも非常に高い	△ サーバ側の構成変更は容易だがクライアント側は変更が困難	○ 要件に応じた構成変更が容易

図Ⅱ-3-1.　集中システムと分散システム

【解説】

1) 集中システム

集中システムとは、メインフレームと呼ばれる大型コンピュータで、データの処理を集中的に行うシステムであり、1980 年代までは情報システムの主流となっていた。現在でも信頼性が求められる分野では引き続き用いられている。

* 集中システムのメリット
  - 高度な運用管理、セキュリティ管理の仕組みがあらかじめ用意されている。
  - 一般に信頼性が高く、高負荷時の性能に優れる。
* 集中システムのデメリット
  - ハード・ソフト両面に高い耐障害性が求められるため、導入コストが非常に高い。

2) 分散システム

分散システムとは、ネットワークを通じて複数のコンピュータが協調して処理を行うシステムであり、現在主流となっている方式である。

* 分散システムのメリット
  - 集中システムと比較して導入コストが低い。
  - 耐障害性を持たせる、処理を高速化する、といった目的に応じた構成をとることが可能であり、集中システムと比較して拡張が容易である。
* 分散システムのデメリット
  - 複数のコンピュータを管理するため、保守やセキュリティ、運用の管理が複雑になる。

3) クライアントサーバシステム

クライアントサーバシステムは、特定の処理を集中的に担当するサーバソフトウェアとユーザが使用するコンピュータ上のクライアントソフトウェアが協調して処理を行う分散システムであり、特にクライアントソフトウェアでビジネスロジックを担当し、サーバソフトウェアがデータ管理などを行う方式がかつての業務システムでは主流であった。

* クライアントサーバシステムのメリット
  - 処理をクライアントとサーバで分担することにより、負荷の集中を軽減できる。
  - 専用のクライアントソフトウェアにより、ユーザフレンドリなインタフェースを実現できる。
* クライアントサーバシステムのデメリット
  - 専用ソフトウェアを各クライアントマシンにインストールする必要があり、管理が煩雑となる。

4) Web システム

Web システムはクライアントサーバシステムの構成要素の内、クライアントソフトウェアとして Web ブラウザを用いるシステムであり、現在普及が進んでいる。

* Web システムのメリット
  - クライアントマシンに専用ソフトウェアをインストールする必要がなく、管理が容易である。
* Web システムのデメリット
  - UI がブラウザの機能に限定されるため、操作性が犠牲になる場合がある。ただし近年は Ajax などの利用により、専用ソフトウェアと同等の使い勝手を実現する例も増えている。

スキル区分	OSS モデルカリキュラムの科目	レベル
基礎分野	3　コンピュータシステムやアーキテクチャに関する知識 II	応用
習得ポイント	II -3-2. 非機能要件とシステム構成	
対応する コースウェア	第 9 回 (コンピュータシステムの構成)	

II -3-2. 非機能要件とシステム構成

コンピュータシステムを構成する非機能要件について述べる。システム構成で要点となるクラスタ構成や、ネットワーク負荷分散、サーバ負荷分散といった負荷分散構成、データ分散構成によるセキュリティ構成などについて説明する。

【学習の要点】

* 非機能要件とは機能要件以外の要件を指し、信頼性や効率性などに代表される。
* 非機能要件を満たすためには、システム構成を適した形にするよう考慮する必要が生じる場合がある。
* クラスタ構成により信頼性向上、負荷分散構成により効率性向上、データ分散構成によりセキュリティ向上を見込める場合がある。

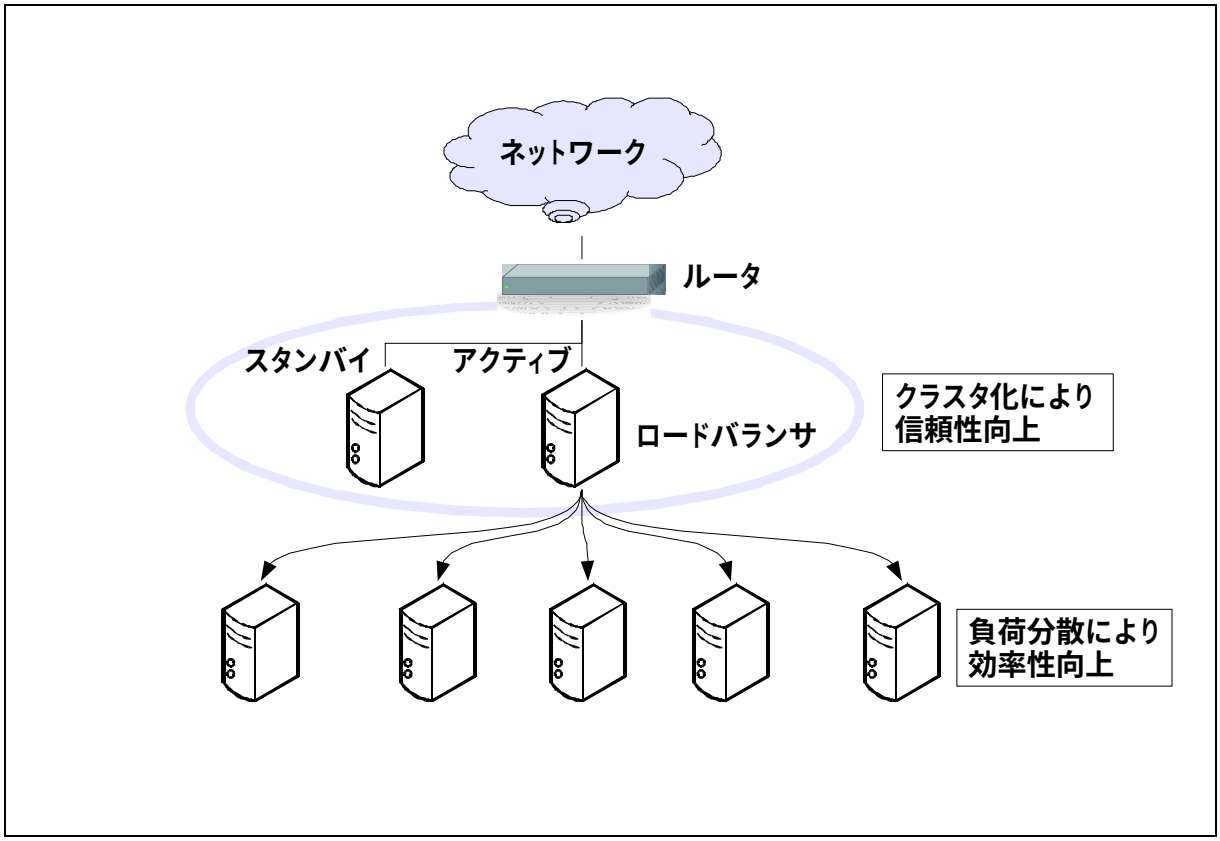


図 II -3-2.　非機能要件を実現するシステム構成

【解説】

1) 非機能要件

* 非機能要件とは機能要件以外の要件を指し、信頼性、効率性、保守性、使用性などが含まれる。
* 非機能要件の管理にあたってはISO9126(ソフトウェア品質特性)を参考にすることができる。

2) 非機能要件を実現するシステム構成

非機能要件を実現するためには、アプリケーション側での対応に加え、適切なシステム構成を適用することも必要となる。

* クラスタ構成による高信頼性の実現
  - システムをアクティブ系とスタンバイ系の 2 系統に冗長化し、アクティブ系に障害が発生した場合、ただちにスタンバイ系に切り替える構成にすることで、信頼性の向上が期待できる。
* ネットワーク・サーバ負荷分散による高効率性の実現
  - 処理を行うサーバを複数台用意し、トラフィックおよび処理の分散を行う構成にすることで、効率性の向上が期待できる。
* データ分散構成によるセキュリティの実現
  - OSS として提供されている分散ファイルシステム Gfarm を利用して情報を断片化して管理することにより、セキュリティ強化を実現することも可能である。顧客情報の管理に利用した事例がある。

3) OSS による高信頼・高効率システムの構成例

高信頼性・高効率性を実現するシステム構成の代表例として、冗長化されたロードバランサと、複数の実サーバの 2 層によるシステム構成があげられる。OSS を利用することによってこのような構成をとることが可能である。

* LVS (Linux Virtual Server)と Ldirectord による負荷分散
  - 実サーバ群に対し負荷分散を行うロードバランサとして、LVS を利用することができる。
  - Ldirectord を利用して実サーバ上のサービスを監視することにより、不適切な状態の実サーバを LVS による処理の分散対象から外す処理を自動化することができる。
* Heartbeat によるクラスタ構成
  - Heartbeat を利用することでアクティブ系とスタンバイ系の 2 系統による冗長化を実現できる。
  - LVS と Ldirectord を利用したロードバランサによる負荷分散と、Heartbeat によるロードバランサ自体のクラスタ化を併用することで、高信頼性と高効率性の両方を実現した構成をとることが可能である。

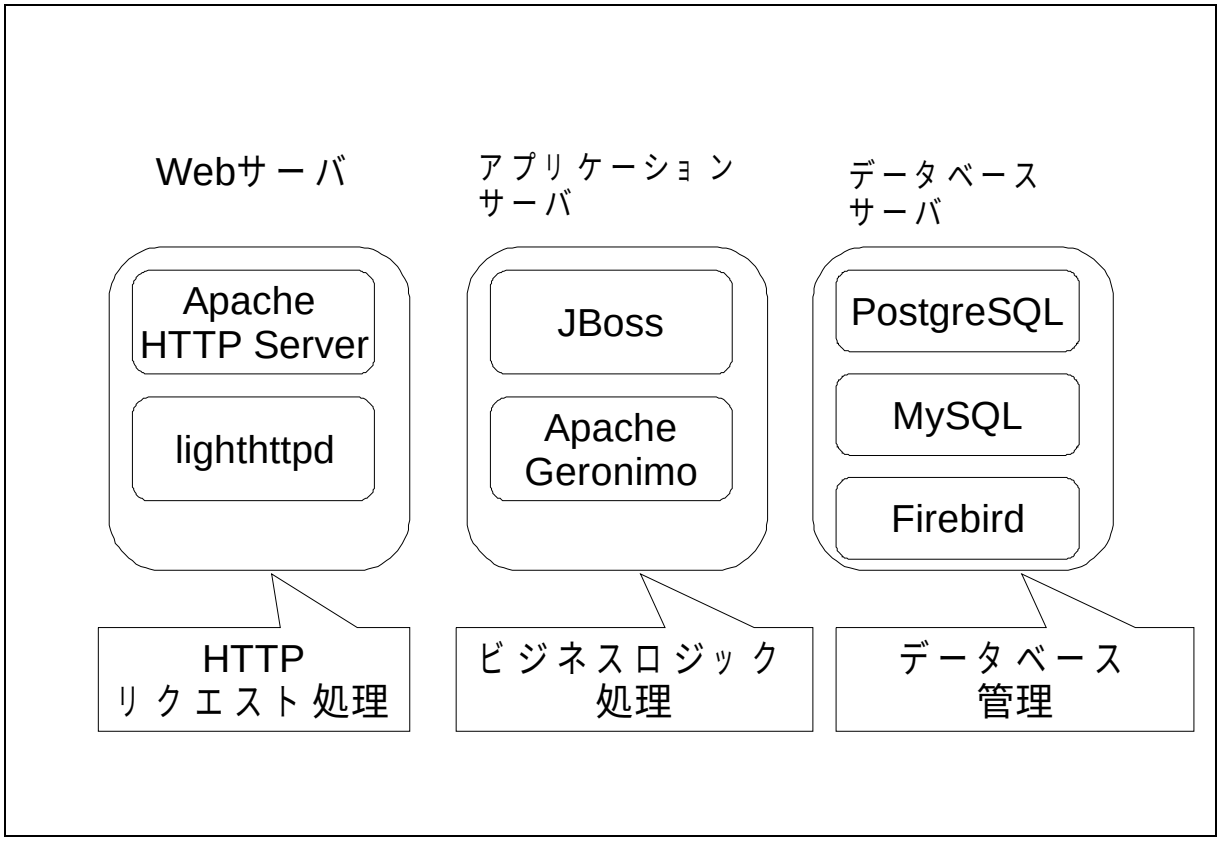
スキル区分	OSS モデルカリキュラムの科目	レベル
基礎分野	3　コンピュータシステムやアーキテクチャに関する知識 II	応用
習得ポイント	II -3-3. OSS によるシステムアーキテクチャの実現	
対応する コースウェア	第 9 回 (コンピュータシステムの構成)	

II -3-3. OSS によるシステムアーキテクチャの実現

代表的なコンピュータシステムの例として、Web サーバ、アプリケーションサーバ、データベースサーバなどがある。これらについて、どのような OSS を用いて構成すると効果的に実現できるかを、具体的な例を用いて示す。

【学習の要点】

* Web サーバ、アプリケーションサーバ、データベースサーバなど、コンピュータシステムを構成する主要な要素の多くは OSS を利用することができる。
* コンピュータシステムの構成要素毎に複数の実装が存在するため、実現したいシステムに応じて適した組み合わせを選択する必要がある。



図Ⅱ-3-3.　システム基盤に用いられる OSS

【解説】

1) OSS によるシステムアーキテクチャの実現

Web サーバ、アプリケーションサーバ、データベースサーバなど、コンピュータシステムを構成する主要な要素の多くは OSS を利用することが可能であり、それぞれ複数の実装が存在する。

2) Web サーバ

Web サーバはクライアントからの HTTP リクエストに応じて適切な HTML や画像データを返却する機能を持ったソフトウェア、もしくはその機能を提供するサーバコンピュータである。

* システムアーキテクチャの一部としての Web サーバは、クライアントと直接通信を行い、リクエストの内容に応じて、アプリケーションサーバなど適したコンポーネントに処理を振り分ける役割を持つ。
* OSS の Web サーバとして広く利用されている Apache HTTP Server は、システムアーキテクチャの一部として利用される例も多く、多くのアプリケーションサーバでは Apache HTTP Server と連携する手順についてのドキュメントが提供されている。

3) アプリケーションサーバ

アプリケーションサーバはビジネスロジックなどをサーバサイドで実行する環境を提供するソフトウェア、もしくはその機能を提供するサーバコンピュータである。

* 多くのアプリケーションサーバはシステムアーキテクチャの一部として利用されることを前提としており、Web インタフェースは Web サーバとの連携により提供され、データの永続化サービスはデータベースサーバとの連携により提供されることが多い。
* OSS のアプリケーションサーバの多くは、OSS の Web サーバおよびデータベースサーバとの連携手順についてのドキュメントを提供している。
* OSS のアプリケーションサーバとして広く利用されている JBoss では、Apache HTTP Server、および MySQL と連携する場合の手順が公開されている。

4) データベースサーバ

データベースサーバはデータベースを保持し、外部のアプリケーションに対して、データの永続化サービス、および検索、抽出などのインタフェースを提供するソフトウェア、もしくはその機能を提供するサーバコンピュータである。

* システムアーキテクチャの一部としてのデータベースサーバは、アプリケーションサーバなど、データに対する加工処理を行うソフトウェアからアクセスされ、処理対象のデータの格納、および検索、抽出などのインタフェースの提供を行う。
* OSS のデータベースサーバとして広く利用されている PostgreSQL、MySQL、Firebird などは、システムアーキテクチャの一部として利用される例も多く、外部のソフトウェアからアクセスするためのドライバが提供されている。

スキル区分	OSS モデルカリキュラムの科目	レベル
基礎分野	3　コンピュータシステムやアーキテクチャに関する知識 II	応用
習得ポイント	II -3-4. Web システムのアーキテクチャ	
対応する コースウェア	第 10 回 (システムアーキテクチャの活用事例)	

II -3-4. Web システムのアーキテクチャ

Web システムの事例を対象として、システムアーキテクチャ検討における OSS 活用のポイントについて解説する。システムアーキテクチャ導入時のシナリオにおいて検討すべき項目や、実際に OSS がどのように活用されてシステム基盤を構築するかについて述べる。

【学習の要点】

* Web システムを OSS を利用して構築する際には機能性に加え、効率性、信頼性、情報入手の容易性、サポートの必要の有無等について特に検討が必要となる。
* 一般的な Web システムでは、http リクエスト・レスポンスを処理する機能、ビジネスロジックを処理する機能、データを保存する機能などが必要となる。
* OSS を組み合わせることで、Web システムに必要な機能の多くを満たすことができる。

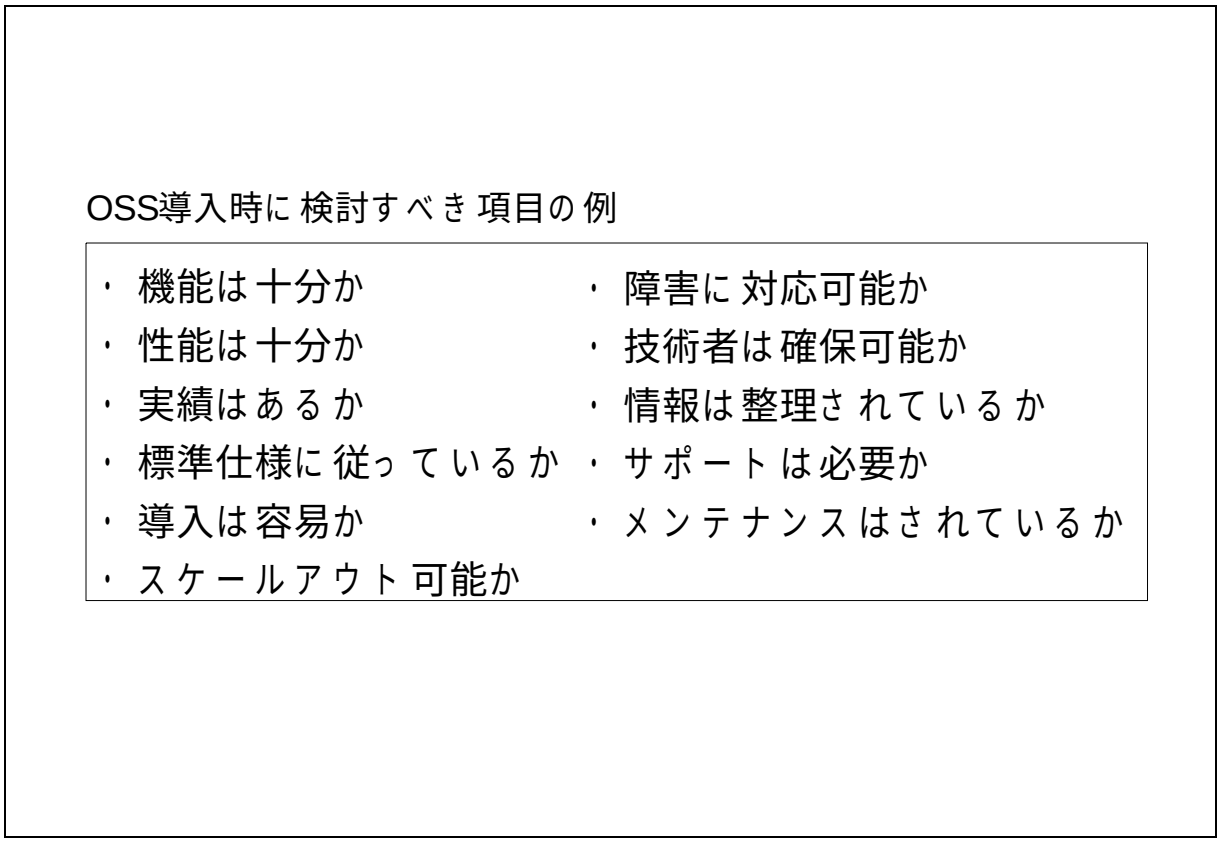


図 II -3-4. OSS 導入時に検討すべき項目の例

【解説】

1) OSS 導入に際し、検討すべき項目

Web システムにおいて OSS を導入する場合に検討すべき項目の代表例を以下に挙げる。

* 機能性

  OSS はプロプライエタリなソフトウェアと比較して機能が少ない場合がある。そのため、システムに必要な機能を明確にするとともに、OSS が必要な機能を提供しているかどうかを調査する必要がある。広く用いられている OSS に関しては、OSS の Web サイト、または競合する製品を扱う企業の Web サイトから機能比較に関する情報が提供されている場合も多い。

* 効率・信頼性

  OSS は効率性、信頼性の面でプロプライエタリなソフトウェアに及ばない場合がある。そのため、想定されるユーザ数やリクエスト数に対し、十分な性能が発揮できるかなどについてあらかじめ調査、検証が必要である。また、障害発生時の復旧機能や、復旧手順の容易さなどについてもあらかじめ調査するべきである。

* 情報入手の容易さ

  一般的に OSS はプロプライエタリなソフトウェアに比べてドキュメントが不足している場合が多く、特にビギナ向けのドキュメントや、トラブル時の対処に関するドキュメントが少ない場合が多い。そのため、管理者の必要とする情報が容易に入手可能かどうかについて、あらかじめ調査しておく必要がある。

* サポート

  OSS の中には有償で企業によるサポートが提供されるものや、無償版とは別に機能の異なる有償版が提供されるものがあり、サポートを利用するかどうかについても検討する必要がある。

* メンテナンス

  メンテナンスが継続して行われているかどうかも調査するべきポイントとなる。更新が行われていない OSS はセキュリティ関連の不具合も放置されている場合がある。

2) OSS による Web システム

一般的な Web システムでは、http リクエスト・レスポンスを処理する機能、業務ロジックを処理する機能、データを保存する機能などが必要となる。OSS を組み合わせることで、必要な機能の多くを満たすことができる。

* http リクエスト・レスポンスを処理する機能

  OSS の Web サーバとして Apache HTTP Server が広く用いられている他、Tomcat、JBoss など Web サーバ機能を持つアプリケーションサーバも存在する。

* ビジネスロジックを処理する機能

  CGI や PHP など Web サーバから直接起動されるプログラムによってビジネスロジックを処理することができる。また、セキュリティや複数のリクエスト間のトランザクションなど、高度な機能を提供するアプリケーションサーバとして JBoss などを利用することも可能である。

* データ保存機能

  データベースサーバとして、MySQL、PostgreSQL などを利用することができる。より簡易な手段として、データをファイルとして保存する SQLite などを利用することも可能である。

スキル区分	OSS モデルカリキュラムの科目	レベル
基礎分野	3　コンピュータシステムやアーキテクチャに関する知識 II	応用
習得ポイント	II -3-5. OSS による Web3 層アプリケーション	
対応する コースウェア	第 11 回 (Web システムのアーキテクチャ)	

II -3-5. OSS による Web3 層アプリケーション

Web3 層アプリケーションとは何か解説し、Web アプリケーション開発の特徴や利点について説明する。そのうえで、OSS を活用した 3 層アプリケーションの具体的な構築例を示し、Web3 層アプリケーション開発における OSS の効果的な利用方法を解説する。

【学習の要点】

* Web3 層アプリケーションとは、Web システムの構成要素をプレゼンテーション層、アプリケーション層、データ層の 3 層に分割し、独立したモジュールとして設計するアプリケーションである。
* 3 層アーキテクチャを採用することにより、ユーザインタフェース、ビジネスロジック、データベースの相互依存を抑え、変更容易性、拡張性を高めることができる。
* Web3 層アプリケーションはプレゼンテーション層を担当する Web サーバ、ビジネスロジックを担当するアプリケーションサーバ、データ層を担当するデータベースサーバによって構築される場合が多い。

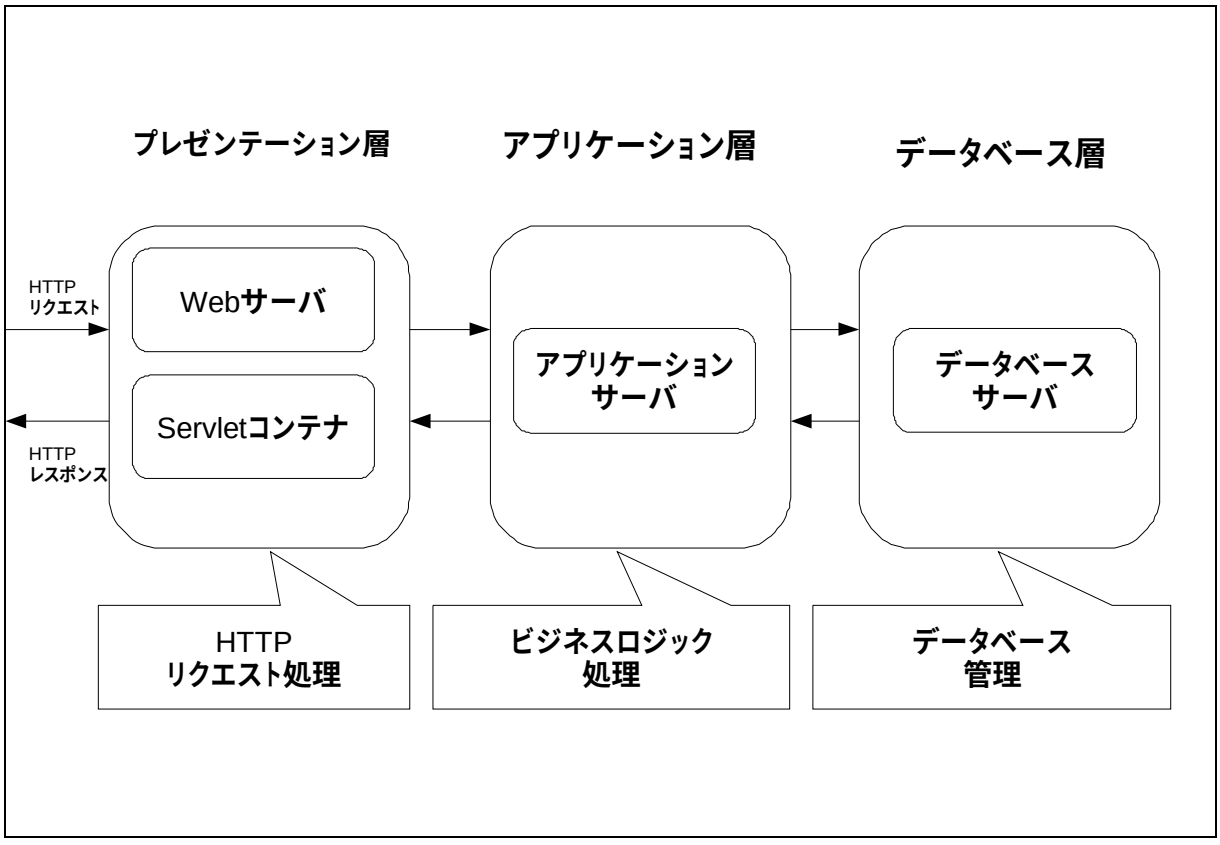


図 II -3-5. OSS による Web3 層アプリケーション

【解説】

1) Web3 層アプリケーション

Web3 層アプリケーションとは、Web システムの構成要素をプレゼンテーション層、アプリケーション層、データ層の 3 層に分割し、独立したモジュールとして設計するアプリケーションである。多くの場合、3 層はそれぞれ別のハードウェア、別の OS で動作可能となるように設計され、実際にそのように運用される場合が多い。

* Web3 層アーキテクチャのメリット
  3 層を独立したモジュールとして設計することにより相互依存性が抑えられるため、要求に応じた機能変更、拡張が容易となる。例えば、データベースの性能が問題となった場合に、データベースサーバのクラスタ化や、OSS からプロプライエタリなデータベースサーバ製品への変更などを行う場合でも、他の部分に対する影響を少なくすることができる。

2) 各層の役割と構成要素

* プレゼンテーション層
  ブラウザからの HTTP リクエストを処理し、必要に応じてアプリケーション層に処理を委譲する。また、アプリケーション層から処理結果を受け取り、ブラウザに返す。Web サーバにより構成され Apache HTTP Server などを利用することができる。
* アプリケーション層
  プレゼンテーション層からの要求に応じてビジネスロジックを処理する。必要に応じてデータ層にアクセスし、データの検索、保存などを行う。アプリケーションサーバにより構成され JBoss などを利用することができる。
* データ層
  データを管理し、アプリケーション層に対してデータの検索、抽出、保存などのインタフェースを提供する。データベースサーバにより構成され PostgreSQL、MySQL などを利用することができる。

3) ソフトウェアの 3 層アーキテクチャ

プレゼンテーション層、アプリケーション層、データ層の 3 層に分割する手法は、システムアーキテクチャだけでなく、ソフトウェアアーキテクチャにも適用される場合がある。この場合、3 層はそれぞれ同一のソフトウェア内のモジュールを指すため、前述の分類とは異なる場合がある。以下に Java EE の場合の例を挙げる。

* プレゼンテーション層
  リクエストの処理とユーザインタフェースの提供を行うモジュールであり、Servlet や JSP によって実装される。
* アプリケーション層
  ビジネスロジックを処理するモジュールであり、Java Bean や EJB の Session Bean によって実装される。
* データ層
  ソフトウェアによって扱われるデータを表現したモジュールであり、Java Bean や EJB の Entity Bean によって実装される。

スキル区分	OSS モデルカリキュラムの科目	レベル
基礎分野	3　コンピュータシステムやアーキテクチャに関する知識 II	応用
習得ポイント	II -3-6. OSS によるシステム基盤設計の手順	
対応する コースウェア	第 12 回 (OSS を活用した基盤設計ケースワーク) 第 13 回 (オープンソースシステムアーキテクチャ構築)	

II -3-6. OSS によるシステム基盤設計の手順

OSS を活用してシステム基盤を構築する手順を理解させる。OSS プロダクトの選定方法からシステムを構成するソフトウェアの配置、ハードウェア性能の検討、非機能要件検討といった具体的な手順を概説する。

【学習の要点】

* OSS によるシステム基盤設計の際には、機能要件、非機能要件を明らかにし、要件を満たすことのできるソフトウェアを選定する必要がある。
* 非機能要件の検討にあたっては、ソフトウェアだけでなく、ハードウェアの採りうる構成も含めて考慮する必要がある。

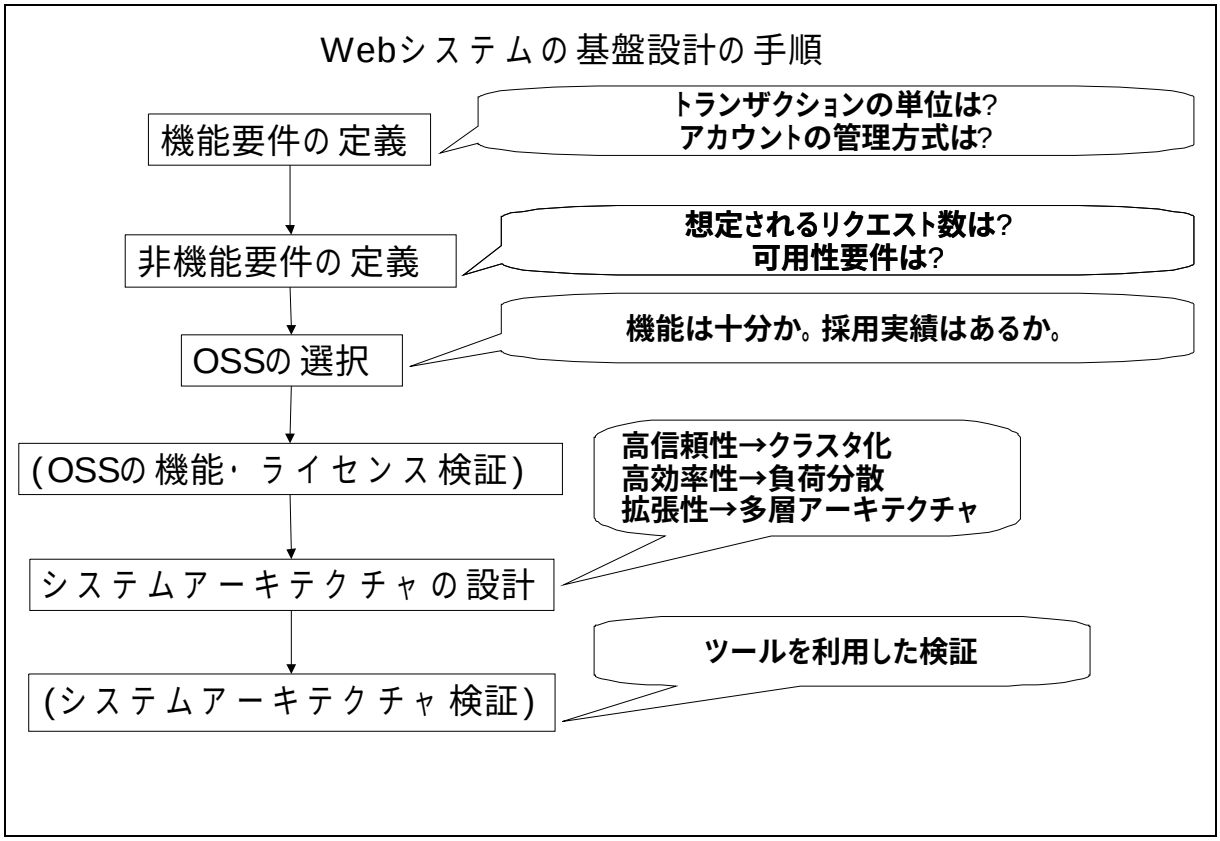


図 II -3-6. OSS によるシステム基盤設計の手順

【解説】

1) OSS によるシステム基盤設計手順

システム基盤設計の手順とポイントを以下に挙げる。

* 機能要件の定義

システムに要求される機能を明確にする。要求される機能により、利用するソフトウェアや技術を決定する。機能要件のうち、システム基盤設計に影響を与える要素の例を以下にあげる。

- セキュリティ機能

暗号化されるべきデータや、ユーザ認証、アクセス制御などについて仕様を明確にし、情報が適切に保護されるアーキテクチャを設計する必要がある。

- トランザクション機能

トランザクションの発生するオペレーション、トランザクションの単位などを定義し、適切なトランザクションの管理方法を設計する必要がある。

* 非機能要件の定義

非機能要件は、システム基盤の設計に直接影響を与える。そのため、チェックリストなどを用いて要求項目を漏れなく明確にしておく必要がある。考慮すべき項目の例を以下に挙げる。

- 性能要件

想定されるリクエスト数などを考慮する。

- 信頼性要件

可用性要件のほか、障害発生時のデータ復旧要件なども考慮する。

- 運用要件

運用のオペレーションや、監視要件、障害通知要件なども考慮する。

* OSS の選定

機能要件、および非機能要件を満たす OSS を選定する。

- OSS の機能についての情報は、OSS の開発 Web サイトから得られる場合が多い。
- OSS の適用例に関する情報として OSS iPedia(http://ossipedia.ipa.go.jp)を利用することができる。

* システムアーキテクチャの設計

機能要件、および非機能要件を満たすアーキテクチャの設計を行う。例を以下に挙げる。

- システムに高信頼性が求められる場合、クラスタ化による冗長設計を行う。
- システムに高効率性が求められる場合、負荷分散を行う。
- システムにスケーラビリティが求められる場合、複数の層に分割した設計にすることにより、将来の拡張に備える。

2) 非機能要件の検証

非機能要件の検証にあたっては OSS のツールを利用できる場合がある。例として、日本 OSS 推進フォーラム開発基盤ワーキンググループより手法、およびツールが公開されている。
(http://www.ipa.go.jp/software/open/forum/development/index.html)

スキル区分	OSS モデルカリキュラムの科目	レベル
基礎分野	3　コンピュータシステムやアーキテクチャに関する知識 II	応用
習得ポイント	II -3-7. OSS によるシステム基盤設計事例	
対応する コースウェア	第 12 回 (OSS を活用した基盤設計ケースワーク) 第 13 回 (オープンソースシステムアーキテクチャ構築)	

II -3-7. OSS によるシステム基盤設計事例

「OSS によるシステム基盤設計の手順」において示した手順を具体的な事例として示す。要件に適したソフトウェアの選定やシステムアーキテクチャの設計など、実際にシステム基盤を設計する際に必要となる知識について解説する。

【学習の要点】

* セキュリティ要件の実現例として、SSL による通信の暗号化が Apache HTTP Server や、PostgreSQL、MySQL など、様々な OSS でサポートされている。
* トランザクション管理機能など、ビジネスアプリケーションに必要な機能の多くがアプリケーションサーバによって提供されている。
* 非機能要件を実現するための負荷分散機能、冗長化機能は、データベースサーバなど、サービスを提供するソフトウェア自身の機能や、拡張機能として提供されている場合がある。

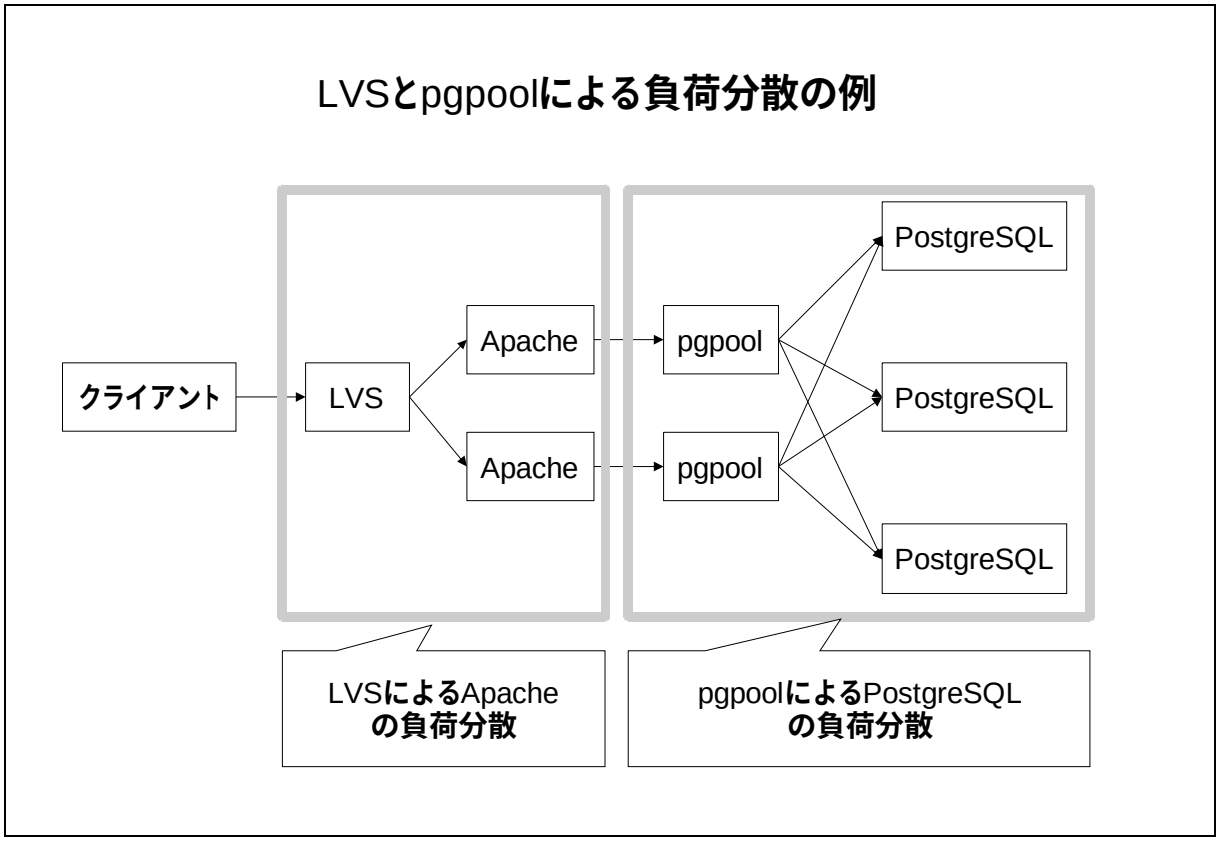


図 II -3-7. 負荷分散構成の例

【解説】

1) 機能要件の実現

機能要件の実現を支援するソフトウェア、アーキテクチャの例を以下にあげる。

* セキュリティ機能

Web システムの場合、認証の際のパスワードなど、データの暗号化が求められる場合がある。Web システムにおける暗号化技術としては SSL による通信が広く用いられており、Apache HTTP Server などの Web サーバや、MySQL や PostgreSQL などのデータベースサーバなどで利用することができる。

* トランザクション機能

JBoss など、Java EE 仕様に準拠したアプリケーションサーバは、開発者によるトランザクション制御の他に、サーバによるトランザクション制御機能を提供している。また、分散トランザクションなど、高度なトランザクションを実現する OSS として「JBoss Transactions」などが公開されている。

* アカウント管理機能

Web システムのうち、ユーザからのリクエストによってアカウントの作成を行うシステムの場合、アカウントの不正取得を防ぐ仕組みが必要となる。広く利用されている仕組みとしては、アカウント作成時に E メールアドレスを入力させ、アカウントの活性化を行うための URL を別途 E メールとして送付する方法がある。Web アプリケーション向けのフレームワークでは、このようなアカウント管理を容易に実装するための機能が提供されている場合がある。

2) 非機能要件の実現

非機能要件の実現を支援するソフトウェア、アーキテクチャの例を以下にあげる。

* 性能要件

Web システムの場合、クラスタリングによる負荷分散や、コンテンツのキャッシュ機能の利用によって応答速度を向上させることが可能である。クラスタリングによる負荷分散を実現する OSS として LVS があげられる。詳細は「II-3-2. 非機能要件とシステム構成」を参照のこと。コンテンツのキャッシュ機能を実現する OSS としては、Squid などがある。また、多くのデータベースサーバでは、データを複数のサーバ間でレプリケーションし、負荷分散、および冗長化する機能が提供されている。例としては MySQL で標準機能として提供されているレプリケーション機能や、PostgreSQL で利用可能な pgpool などがあげられる。

* 信頼性要件

信頼性を向上させる方法としては、システムの冗長化が広く用いられている。サーバの冗長化を実現する OSS として Heartbeat があげられる。詳細は「II-3-2. 非機能要件とシステム構成」を参照のこと。

* 運用要件

安定した運用を継続する上で必要となるサービスの監視や、障害通知を行う OSS として Nagios や Hobbit などが利用されている。また、SNMP を利用したネットワーク機器の監視を行うソフトウェアとして MRTG などが利用されている。

スキル区分	OSS モデルカリキュラムの科目	レベル
基礎分野	3　コンピュータシステムやアーキテクチャに関する知識 II	応用
習得ポイント	II -3-8. OSS が動作するハードウェア	
対応する コースウェア	第 14 回 (OSS の動作環境としてのハードウェア)	

II -3-8. OSS が動作するハードウェア

OSS が動作する基盤となるハードウェアについて、サーバ、クライアントといったそれぞれ利用環境に合わせたハードウェアの特徴を整理する。さらに、OSS を動作させるにあたって留意すべきポイントや課題についても述べる。

【学習の要点】

* OSS を動作させるサーバのハードウェアの選定にあたっては、使用する基本ソフトをハードウェアベンダがサポートしていることが望ましい。
* サーバのハードウェアのスペックの検討にあたっては、動作させるソフトウェア、想定されるアクセス数、およびユーザサイドからの性能要件などを考慮する必要がある。
* ハードウェアの選定に関するポイントとしては、使用するソフトウェアが動作可能であることや仕様が公開されていることを確認する必要がある。

図 II -3-8. OSS を動作させるハードウェアの特徴

【解説】

1) サーバ用ハードウェアの選定

サーバ用ハードウェアに求められる要件は用途によって様々であり、複数のネットワークインタフェースを扱う、特別なストレージデバイスを扱う、などが想定される。また、信頼性など、非機能要件に対応するため、各部品が冗長化された製品や、OS がダウンした場合でもネットワーク経由でハードウェアの制御が可能な製品などが存在する。さらに、サーバ用ハードウェアの場合、必要な要件がユーザ数など外部の条件によって変動する点に注意が必要である。

* 動作に必要となるハードウェア機能

オープンソース OS の各ハードウェアへの対応状況はそれぞれの OS によって異なるため、対応状況については各オープンソース OS の Web サイトや、ハードウェアベンダの Web サイト上で確認する必要がある。サーバ機において特に注意が必要な項目として、使用するストレージデバイスやネットワークインタフェースへの対応などが挙げられる。

* ハードウェアベンダによる OS のサポート

ビジネス用途のサーバにオープンソース OS をインストールして利用する場合、ハードウェアベンダによりその OS の動作が保証されていることが望ましい。サポートされていない OS をインストールした場合、不具合があってもユーザ責任になってしまう、というだけでなく、ハードウェアの機能・性能の全てを活用できない可能性がある。また、ラックマウントサーバやブレードサーバなど、サーバ専用機として設計されたハードウェアは専用の管理ソフトウェアが用意されていることが多いが、これを活用できるのはベンダによって動作が保証されている OS のみである場合がある。多くのハードウェアベンダにより動作が保証されているオープンソース OS としては Red Hat Enterprise Linux が挙げられる。また、Red Hat Enterprise Linux と互換性が高い CentOS を代替として用いることが可能な場合もある。

* ハードウェアのスペックの検討

性能要件や、扱うデータ量などの各要件を元にハードウェアのスペックを検討する。要件を満たすために必要なハードウェア性能については、事例を元に決定することが最も確実性が高いが、Web 上で公開されているベンチマーク結果などのリソースを参考にできる場合がある。

2) クライアント用ハードウェアの選定

クライアント用ハードウェアの選定に関するポイントを述べる。

* 動作に必要となるハードウェア機能

クライアントでは、基本機能に加え、使用するソフトウェアの要件に応じてグラフィックカード、サウンドカード、あるいは指紋認証デバイスや、IC カードリーダなどの対応が問題となる場合がある。クライアント用途で用いられるハードウェアはデバイスの多様性が高く、ソフトウェア側での対応が難しいため、対応状況について、各オープンソース OS の Web サイトや、ハードウェアベンダの Web サイト上で確認する必要がある。

* ハードウェアのスペックの検討

クライアントのハードウェアに求められるハードウェアのスペックは、クライアント上で動作させるソフトウェアが必要とするスペックによって決定することができる。サーバと異なり、ユーザ数などの外部の要因によって左右されることは少ない。

スキル区分	OSS モデルカリキュラムの科目	レベル
基礎分野	3　コンピュータシステムやアーキテクチャに関する知識 II	応用
習得ポイント	II -3-9. コンピュータ以外への発展	
対応する コースウェア	第 15 回 (これからのオープンソースアーキテクチャの動向)	

II -3-9. コンピュータ以外への発展

当初は一般のコンピュータが OSS の主たる適用対象であったが、現在では携帯電話や PDA、情報家電といった組み込み分野など様々なアーキテクチャへと対象範囲が拡大している。これらに関して、アプリケーション基盤の構成要素や設計のポイントなどについて説明する。

【学習の要点】

* 携帯電話や PDA、情報家電といった組み込み分野など、一般のコンピュータ以外の分野においても Linux をはじめとした OSS が利用されている。
* 一般のコンピュータ以外のハードウェアで OSS を動作させる場合、リソースや使用できるライブラリの制限を考慮する必要がある。
* Linux におけるリアルタイム性の実現など、組み込み用途に適した特長を持つ OSS の開発も行われている。

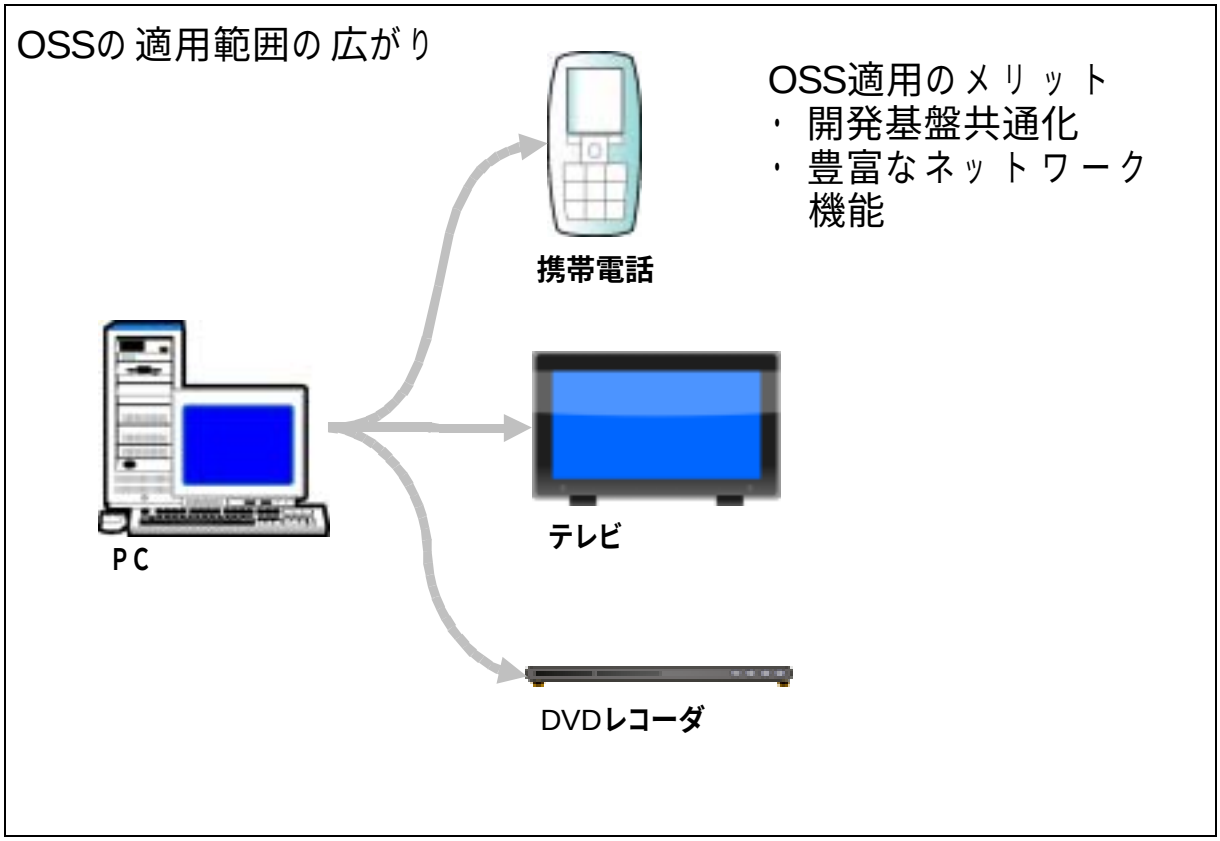


図 II -3-9. OSS の適用範囲の広がり

【解説】

1) OSS の適用範囲と構成要素

一般のコンピュータ以外の分野においてもアプリケーション基盤の構成要素として Linux をはじめとした OSS が利用されている。

* 携帯電話･PDA
  携帯電話･PDA の基本ソフトとして Linux が広く利用されている。また、Linux ベースの携帯電話･PDA では GUI ツールキットとして同じく OSS の Qt が利用される場合も多い。2008 年 7 月現在の動きとして、携帯電話の基本ソフトとして最も利用率の高い Symbian OS のオープンソース化が発表されている。さらに、Linux をベースとした新たなモバイル機器用ソフトウェア基盤として Android が発表され、注目を集めている。
* その他の機器
  各種ネットワーク機器やデジタル家電の多くの製品で Linux をはじめとした OSS が利用されている。

2) 一般のコンピュータ以外のハードウェアにおける OSS

一般のコンピュータ以外のハードウェアにおいて、Linux のようなオープンな仕様のソフトウェアをアプリケーション開発の基盤として用いることにより、TCP/IP によるネットワーク機能に代表される豊富な機能を利用できる他、アプリケーションの仕様や開発方法を共通化できるというメリットがある。例えば OS として Linux が搭載された機器では、通常の PC と同様の開発方法を採ることが可能な場合も多い。しかしながら、共通のアプリケーション開発基盤を利用した場合でも、ハードウェア毎に使用できるリソースやライブラリの制限が存在することがほとんどであり、開発の際にはこれらの制限を考慮する必要がある。

3) 組み込みに特化した OSS

Linux におけるリアルタイム性の実現など、組み込み用途に適した特長を持つ OSS の研究･開発が継続的に行われている。

* CE Linux フォーラム(http://www.celinuxforum.org)では、家電や携帯電話を対象にした Linux の機能強化･普及促進を目的として様々な活動が行われている。Web サイトでは、組み込み Linux に関する技術情報や、消費者向け電気製品で利用される Linux に対する要求仕様、および Linux カーネルのパッチなどが公開されており、その成果を利用できる。
* リアルタイム性やリソースが限られた状況での動作など、組み込み用途向けの機能を実現した Linux ディストリビューションとして MontaVista Linux、Wind River Linux などが挙げられる。

スキル区分	OSS モデルカリキュラムの科目	レベル
基礎分野	3　コンピュータシステムやアーキテクチャに関する知識 II	応用
習得ポイント	II -3-10. 次世代ネットワークアーキテクチャ	
対応する コースウェア	第 15 回 (これからのオープンソースアーキテクチャの動向)	

II -3-10. 次世代ネットワークアーキテクチャ

OSS と親和性の高いネットワーク技術も、OSS がその基盤を支えているインターネットから、インターネット以外のネットワークへと OSS の適用範囲が広がっている。ここでは、ユビキタスネットワーク、センサネットワークや IPv6 の動向について述べる。

【学習の要点】

* インターネット以外のネットワークに対しても OSS の適用範囲が広がりつつある。
* TRON プロジェクトなど、次世代ネットワークアーキテクチャへの応用が期待される分野においてもオープンソースの実装が存在する。
* Linux をはじめとした OSS の大半は IPv6 に対応済みである。

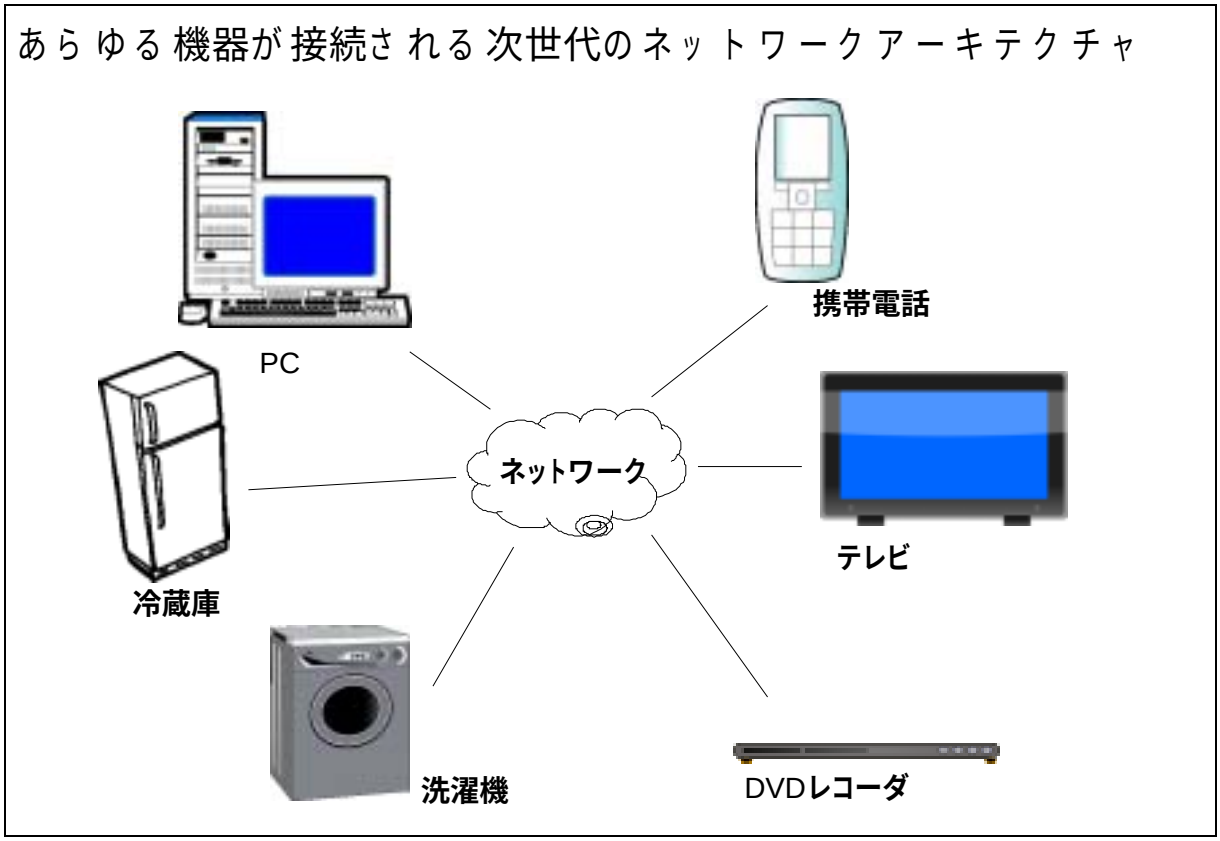


図 II -3-10. 次世代のネットワークアーキテクチャ

【解説】

1) インターネット以外での OSS の利用

インターネット以外のネットワークに対しても OSS の適用範囲が広がりつつある。例えば Asterisk(http://www.asterisk.org)は PBX(構内電話交換機)を実現する OSS であり、市販の PBX と遜色がない機能を提供できるとして注目を集めている。

2) 次世代のネットワークアーキテクチャ

次世代のネットワークアーキテクチャとして、ユビキタスネットワーク、センサネットワークも期待されている。身の回りの様々な機器がネットワークに接続されると共に、機器自身が情報を認識し、さらに機器同士が相互にコミュニケーションをとることができるようになる。これにより、コンピュータを意識せずに利便性の高い生活を送ることができるとされる。

3) TRON プロジェクトと OSS

次世代のネットワークアーキテクチャへの応用が期待される分野として、TRON プロジェクト(http://www.tron.org)があげられる。

* TRON プロジェクトでは、ユビキタスという言葉が一般に使われる以前から「どこでもコンピュータ」をキーワードとして分散コンピューティング環境の実現を目指した取り組みが続けられている。
* TRON プロジェクトの代表的な成果として、組み込みに適したリアルタイム OS の仕様策定がある。
* TRON プロジェクトで策定された仕様に基づいたオープンソース実装として、μITRON 仕様に基づいたカーネルが TOPPERS プロジェクト(http://www.toppers.jp)によって公開されている。
* TRON プロジェクトから派生した T-Engine プロジェクトで開発されたリアルタイム OS「T-Kernel」はソースコードを入手し、改変することができる。

4) IPv6

様々な機器がネットワークに参加する次世代のネットワークアーキテクチャを実現するために必要な技術の 1 つとして、IPv6 があげられることが多い。Linux をはじめとした OSS の大半が IPv6 に既に対応済みであるなど、IPv6 対応が充実していることは OSS の強みの一つであり、将来に向けて OSS の重要性が高まっていく可能性がある。