委託契約書番号：2004 情財第 992 号

2004 年度未踏ソフトウェア創造事業

情報家電プロトコルスタックの開発

成果報告書

開発代表者：今野 賢

担 当 PM：坂村 健

プロジェクト管理組織：（有）枝川建設

平成 17 年 8 月 24 日

目次

1. 要約

2004 年度未踏ソフトウェア創造事業、「情報家電プロトコルスタックの開発」として、UPnP(Universal Plug and Play)[1]を将来の組込みシステムの高性能化に即した高機能版および現状の組み込みシステムに適した軽量版の開発、組み込み系を含む各種プラットフォームへ実装した成果を報告する。

2. 背景及び目的

近年、PC(パーソナルコンピュータ)および DVD レコーダ等のデジタル家電等のネットワーク機器が家庭内に普及して久しい。現在、これらの家庭内ネットワーク相互接続環境を可能にする標準規格として、以前より UPnP が有望視されていたが、家電・PC・モバイル業界等の主要機器メーカーが参加する DLNA(Digital Living Network Alliance)[2]が昨年本格的に組織され、UPnP の AV 機器関連のサービス定義である UPnP/AV[3]をベースとした標準規格によるデジタル家電のガイドラインを策定する等、家庭内ネットワークの相互接続環境の重要性が増している。

また、ユビキタスに代表されるように、近未来的には、デジタル家電だけではなく、あらゆる家庭内機器にネットワーク機能が装備され、更なる相互接続性の確保は必然的でもあるが、それらの主役となるのは現在の PC やデジタル家電のような重厚なネットワーク機器ではなく、より低価格/小サイズである組み込み機器である事が予想される。また近年、PIC や H8 に代表されるようなワンチップマイコンの高性能化/低価格化により、個人レベルでも組み込み機器を開発できる土壌が整備されつつあるが、現状でも組み込み機器開発は個人レベルでは開発環境等の予算的な面で敷居が高いのが実情である。

開発視点から見ると、UPnP は SOAP や SSDP 等の各種プロトコルを用いた要求範囲の広い総合的な仕様であり、UPnP に関連した製品およびプログラムを作成するには、幅広い技術への理解および実装能力が必要となる。(表 1/図 1)

表 1 : UPnP 機能階層/対応プロトコル

UPnP 機能階層	対応プロトコル
アドレス割当	IPv6, AutoIP(IPv4)
デバイス検索	HTTP, HTTPU, SSDP, GENA
サービス確認	HTTP, SOAP
サービス/イベント制御	HTTP, SOAP, GENA

図 1 : UPnP プロトコル階層図

<table>
<tr><td colspan="5">UPnP</td></tr>
<tr><td>SSDP</td><td>GENA</td><td>SSDP</td><td>SOAP</td><td>GENA</td></tr>
<tr><td colspan="2">HTTPMU</td><td>HTTPU</td><td colspan="2">HTTP</td></tr>
<tr><td colspan="3">UDP</td><td colspan="2">TCP</td></tr>
<tr><td colspan="5">IPv4/IPv6</td></tr>
</table>

筆者は以前より UPnP 技術および UPnP/AV に興味があり、個人的に UPnP プロトコルスタックである「CyberLink for C++[4]」および「CyberLink for Java[5]」や、他オープンソースプロジェクトと連携した UPnP/AV サーバ/クライアントである「CyberMediaGate for C++[6]」や「CyberMediaGate for Java[7]」等を開発し、オープンソースとして公開する活動を継続してきた。近年になって、商用を含め UPnP プロトコルスタック提供者は増加しているが、現在でもオープンソースで提供されている C++言語または Java 版の UPnP プロトコルスタックとしては CyberLink が世界唯一のものである。

組み込み系の開発には、将来的なユビキタス時代のプラットフォームとして以前より興味はあったが、筆者の所属する業務内容的には関連性は薄く、個人としても T-Engine のような組み込み系プラットフォームの開発環境を揃えるには経費的に大きな負担であった。それに加え、既存の CyberLink は C++言語および Java で実装され、両者は近年の処理性能向上が著しい PC のような処理性能の高いプラットフォームを暗黙的な前提としており、組み込み系としては処理性能/消費リソース面で動作するように当初から設計されたものではなかった。

上記経緯から、本プロジェクトでは、現状の家庭内ネットワーク相互接続環境の確立、将来のユビキタス時代の到来を見越し、組み込み系プラットフォームを前提にその開発基盤を整備するものとした。

3. プロジェクト概要

情報家電プロトコルスタックとしてUPnPを、組み込み系を含む各種プラットフォームに実装する。将来の組込みシステムの高性能化に即した高機能版と、現状の組み込みシステムに適した軽量版を開発した。

3.1. 高機能版

既存のオープンソースとして開発済みのC++言語版UPnPプロトコルスタック「CyberLink for C++」を高機能版として位置づけた。この高機能版は、UPnP仕様で定義されている全機能が実装済みのものであった。ただし、このプロトコルスタックは、WindowsXPやLinuxのような処理性能の高いプラットフォームを対象として開発されてきた経緯があり、組み込み系プラットフォームでは、より軽量なものが望ましい。本プロジェクトでは、XMLパーサ等の軽量化およびT-Engineを代表とする組み込み系プラットフォームへの移植を主な作業内容とした。

3.2. 軽量版

C++言語およびJava版のUPnPプロトコルスタックの開発経験を活かし、組み込み系プラットフォームを意識した軽量なUPnPプロトコルスタックを開発した。実装はC言語で、高機能版のオブジェクト指向設計を継承しつつ、処理性能および処理サイズに配慮し、UPnP仕様で定義されている全機能を実装した。 便宜的に軽量版と命名しているが、高機能版に対して実装された機能に制限および、遜色がある訳ではない。プログラミング言語/依存するライブラリ自体の軽量性および、基本設計から実装まで一貫して組み込み系プラットフォームを意識した意味合いと理解して頂きたい。

3.3. 対応プラットフォーム

高機能版/軽量版は、それぞれ以下に示すプラットフォームを実装対象とした。(表 2)

表 2: 対応プラットフォーム

プラットフォーム	高機能版	軽量版	補足
Linux	◎	◎	完了
MacOSX	◎	◎	完了
Windows	◎	◎	完了
T-Engine	◎	◎	完了
μITRON	○	○	ネットワーク関連部は動作試験中
BTRON	△	△	プラットフォーム仕様的に実装不可
WindowsCE	×	◎	実施計画書では対応プラットフォーム予定外

BTRON に関しては、実施計画書提出後の調査により、プラットフォームに一部機能不備があり、仕様的に完全な実装は不可能である事が判明した。BTRON に関しては、将来的に開発元に機能拡張に対応して頂くよう PM より働きかけて頂く予定である。

μITRON に関しては、高機能版/軽量版共にプラットフォーム依存部移植含む全ソースの動作確認はほぼ完了しているが、開発環境構築の遅れもあり現状ネットワーク関連部に関して動作確認作業を継続中である。ただしμITRON に関しては、現状利用しているベンダーの開発環境およびサポート体制の問題もあり、今後も PM の支援を受け開発プラットフォームの再選定も視野に入れ進捗予定である。

また、実施計画書では予定外ではあったが、オープンソース公開後の利用者からの要望および作業協力にて、対応プラットフォームとしてマイクロソフト系の組み込み系プラットフォームである WindowsCE を本成果物対象として追加できた。

その他、上記対応プラットフォームには明記していないが、配布パッケージとしてAutomake/Autoconfに対応しているため、その他

UNIXプラットフォームに関しても修正なしで、コンパイル可能なよう配慮した。

4. 事前評価

筆者は組み込み系プラットフォームの開発実務経験に乏しかった為、基本設計作業と並行してプラットフォーム選定を含む組み込み系開発環境に関する評価を最初に実施した。

4.1. 開発環境調査

組み込み系プラットフォームである T-Engine およびμITRON を中心に、本プロジェクトの実行対象となる機種選定の調査を実施した。選定は、これらプラットフォーム仕様では定義されていない、TCP/IP プロトコルスタックおよび同プロトコルスタックのマルチキャスト対応可否を中心に実施した。

4.1.1. T-Engine

当初よりパーソナルメディア株式会社提供の T-Engine 開発キットおよび TCP/IP ライブラリ(PMC T-Shell 開発キット)を中心に開発環境調査を実施した。ただし同社の TCP/IP ライブラリではマルチキャストにて対応していない事が判明。最終的には株式会社エルミックシステム社製の KASAGO for T-Engine の最新版が同機能に対応しており、そのライブラリが提供されている機種 T-Engine/SH7727 開発キットを開発環境として選定した。ただしパーソナルメディア株式会社製の PMC T-Shell 開発キットも、マルチキャスト以外の機能は移植の対象とし、将来同開発キットがマルチキャスト対応時に、最低限の移植で済むように対処した。PMC T-Shell 開発キットに関しては、将来的に開発元に機能拡張に対応して頂くよう PM より働きかけて頂く予定である。

4.1.2. μITRON

トロン協会の ITRON 仕様書を中心に開発環境調査を実施した。その結果、ITRON 仕様範囲(ITRON TCP/IP API 仕様 v1.0)では、T-Engine 同様にマルチキャストに対応していない事が判明。最終的には ITRON 仕様範囲内で独自拡張をして、マルチキャストに対応している株式会社ミスポ製「NORTi Professional」を開発環境として選定した。

4.1.3. BTRON

開発時、唯一の選択肢であったパーソナルメディア株式会社提供の超漢字 4 を選定し、開発環境調査を実施した。しかし BTRON の TCP/IP マネージャ仕様および同社への確認でマルチキャストに対応していない事が判明。T-Engine およびμITRON とは異なり、代替環境がないため一部機能の未実装を前提とし超漢字 4 を開発環境として選定した。超漢字 4 関しては、将来的に開発元に機能拡張に対応して頂くよう PM より働きかけて頂く予定である。

4.2. XML パーサ評価/移植

組み込み向けに軽量かつ高速に動作する XML パーサ処理が必要となり、その評価/移植作業を実施した。当初、既存プロジェクト「CyberX3D for C++[8]」のように bison[9]/flex[10]を構文解析ツールとして、あるいは新規に独自パーサの開発も検討したが、調査段階で既存のオープンソースプロジェクトで組み込み系にも利用できそうな優秀な XML パーサが多いと判断し、これらを選定の中心とした。

従来、高機能版 C++版および Java 版共に Xerces[11]を XML パーサとして採用していたが、ライブラリ規模が大きく、組み込み系プラットフォームでのコンパイル/移植作業に難があった。その為、最終的には高機能版および軽量版共に Expat[12]を XML パーサとして選定した。libxml2[13]はGnome プロジェクト等で採用されており、実績およびドキュメントの豊富さに魅力はあったものの、調査中に XML データ取得時に頻繁なメモリ確保/開放処理が必要となる等が判明、Expat と比較すると軽量感に乏しく、開発当初では選定から除外した。ただし、libxml2 は後のオープンソース公開後のユーザー貢献により、最終的には高機能版/軽量版共に対応を追加した。

4.3. 試験フレームワーク

開発効率の向上を目標に、各プラットフォームにおけるテスト工程を自動化するフレームワークの調査を実施した。C++のフレームワークとしてはCppUnit[14]が有名ではあるが、T-Engine等の組み込み系プラットフォームの機種選定で開発環境整備が遅れていた関係もあり、開発当初はCppUnitを軽量化したCppUnit-x[15]およびCUnit[16]を中心として、試験フレームワークの評価/移植作業をした。

しかしながら、組み込み系プラットフォームの開発環境整備が完了し、CppUnitが機種選定したT-Engineで移植/動作確認がとれた事で、最終的にはCppUnitをテスト工程のプラットフォームとして選定した。

5. 開発内容

5.1. 機能概要

高機能版および軽量版共に、UPnP 仕様で定義されている全機能を実装した。本プロジェクトで実装した機能は、基本機能部、UPnP 基本機能部、UPnP デバイス管理部に大分される。(表 3)

表 3 : 開発機能一覧

大区分	機能	高機能版	軽量版
基本機能部	データ構造	○	○
	文字列操作	○	○
	ネットワーク処理	○	○
	ファイル管理	○	×
	タスク管理	○	○
	時刻管理	○	○
	例外管理	○	×
	リスナー管理	○	○
UPnP 基本機能部	HTTP クライアント処理	○	○
	HTTP サーバ処理	○	○
	XML 処理	○	○
	SOAP 処理	○	○
	SSDP 処理部	○	○
	GENA 処理部	○	○
UPnP デバイス管理部	デバイス管理	○	○
	コントロールポイント管理	○	○

軽量版に関しては、上記ファイル管理および例外管理は実装の対象外とした。前者は組み込み系プラットフォーム等プラットフォームに依ってはファイルシステムが定義されていない、後者は実装言語であるC言語レベルでの例外処理サポートがない為である。

上記開発機能の構成図を以下に示す。(図 2) 基本的にユーザー公開対象および、UPnP 開発者が利用するのはUPnP デバイス管理部の関数群のみとなる。その他、基本機能部およびUPnP 基本機能部に関しては、通常の利用方法では必要性が薄く、今後の開発状況による仕様変更の可能性もあり、マニュアル等の資料では非公開とした。

図 2 : CyberLink 機能階層図

<table>
<tr><td colspan="3">UPnP アプリケーション</td></tr>
<tr><td colspan="2">UPnP デバイス管理部</td><td>デバイス管理</td></tr>
<tr><td colspan="2"></td><td>コントロールポイント管理</td></tr>
<tr><td colspan="3">UPnP 基本機能部</td></tr>
<tr><td></td><td>HTTP クライアント処理</td><td>SOAP 処理</td></tr>
<tr><td></td><td>HTTP サーバ処理</td><td>SSDP 処理</td></tr>
<tr><td></td><td>XML 処理</td><td>GENA 処理</td></tr>
<tr><td colspan="3">基本機能部</td></tr>
<tr><td></td><td>データ構造</td><td>タスク管理</td></tr>
<tr><td></td><td>文字列操作</td><td>時刻管理</td></tr>
<tr><td></td><td>ネットワーク処理</td><td>例外管理</td></tr>
<tr><td></td><td>ファイル管理</td><td>リスナー管理</td></tr>
<tr><td colspan="3">プラットフォーム (OS)</td></tr>
</table>

5.2. 機能詳細

5.2.1. 基本機能部

基本機能部では、マルチプラットフォーム実装の基盤となる各種共通ライブラリを実装の対象であり、ネットワークやスレッド生成等のプラットフォーム依存機能を全て吸収する階層である。本機能部は、今後の開発状況により仕様変更の可能性もあり、基本的にユーザー公開の対象とはしない。

本階層より上位機能部である UPnP 基本機能部/UPnP デバイス管理部については、プラットフォーム依存コードは存在しない。その為、新規プラットフォームに本成果物を移植する際には、本機能部のみが対象となる。ただし、本機能部の全てがプラットフォーム依存/移植の対象ではなく、ネットワーク管理、ファイル管理、タスク管理、時刻管理のみが移植の対象となる。

5.2.1.1 データ構造

各関数オブジェクトの連続したデータ構造を管理する機能を実装した。高機能版では STL の Vecctor コンテナを基本として管理していたが、軽量版では実行速度およびメモリ効率を考慮し、双方向リスト構造での実装を基本とした。(表 4) この機能は、XML のツリー構造は HTTP ヘッダ処理など、オブジェクトが階層的および羅列されたデータを取り扱う、全てのオブジェクト管理に用いられている。

表 4: データ構造機能一覧

No	機能	高機能版	軽量版	補足
1	ベクタ構造	○	×	
2	双方向リスト構造	×	○	

5.2.1.2 文字列操作

文字列をオブジェクト指向的に管理できる文字列管理、文字列トークン管理等の機能を実装した。加えて、ANSI-C 対象外関数の移植性確保、または範囲内であっても仕様上問題のある関数の堅牢性確保のため、該当関数を独自実装した。(表 5)

表 5: 文字列操作機能一覧

No	機能	高機能版	軽量版	補足
1	文字トークン管理	○	○	
2	移植性確保	○	○	strdup 関数等のラッパー関数
3	堅牢性確保	○	○	strncpy 関数等ラッパー関数
4	文字列管理	○	○	文字列をオブジェクトとして管理

5.2.1.3 ネットワーク管理

ネットワーク通信機能、ソケット通信や UDP 通信などの機能を各プラットフォーム固有のライブラリを用いて実装した。(表 6) その他、URL 処理関数等の補助関数も本機能に含まれる。

高機能版は組み込み系プラットフォームを除き IPv6/IPv4 対応、軽量版および組み込み系プラットフォームは IPv4 対応として各機能を実装した。

T-Engine プラットフォームでは株式会社エルミックシステム社製の「KASAGO for T-Engine」、その他プラットフォーム標準のものを TCP/IP プロトコルスタックとして選択した。T-Engine プラットフォーム PMC T-Shell 開発キットおよび BTRON プラットフォームでは、マルチキャスト送受信機不備により動作範囲外であるものの、その他ネットワーク機能については移植の対象とした。

表 6: ネットワーク管理機能一覧

No	機能	高機能版	軽量版	補足
1	ストリームソケット送受信	○	○	
2	UDP ソケット送受信	○	○	
3	マルチキャスト送受信	△	△	BTRON では実装対象外
4	ローカルインタフェース取得	○	○	
5	ネットワーク補助関数	○	○	URL 文字列処理関数等

5.2.1.4 ファイル管理

ファイル入出力や関連ストリーム機能を、各プラットフォーム固有のライブラリを用いて実装した。(表 7) 組み込み系プラットフォームに依ってはファイルシステムが定義されていない場合もあり、軽量版では対象外とした。高機能版についても、T-Engine/μITRON ではファイル関連の機能は実装の対象外とした。

表 7: ファイル管理機能一覧

No	機能	高機能版	軽量版	補足
1	ファイル/パス名	△	×	T-Engine/μITRON では実装対象外
2	ファイル入出力	△	×	T-Engine/μITRON では実装対象外
3	入出力ストリーム	○	×	
4	文字列ストリーム	○	×	
5	ラインストリーム	○	×	

5.2.1.5 タスク管理

タスク管理機能を仮想化するインタフェースを実装した。タスクの生成、同期オブジェクト管理を各プラットフォーム固有のライブラリを用いて実装した。(表 8) タスク生成に関してはスレッド制御、同期オブジェクトに関してはセマフォ機能が実装された機能である。

表 8: タスク管理機能一覧

No	機能	高機能版	軽量版	補足
1	タスク生成	○	○	
2	同期オブジェクト	○	○	

5.2.1.6 時刻管理

時刻管理機能を仮想化するインタフェースを実装した。現在時刻での取得機能を各プラットフォーム固有のライブラリを用いて実装した。(表 9) ただし、μITRON では仕様的に相対時刻の扱いしかないため、指定時刻を UNIX 時刻に変換する仕様とした。

表 9: 時刻管理機能一覧

No	機能	高機能版	軽量版	補足
1	時刻管理	○	○	現在時刻または相対時刻

5.2.1.7 例外管理

例外発生時の共通インタフェースを実装した。軽量版では、実装言語(C 言語)レベルでの対応がない点と、独自実装の工数および実行面で非効率なため、実装対象外とした。(表 10) 高機能版では UPnP デバイス定義エラー時などに例外が発生するが、軽量版では発生せず、関数の返り値で成功/失敗を判定する仕様とした。

表 10: 例外管理機能一覧

No	機能	高機能版	軽量版	補足
1	例外管理	○	×	

5.2.1.8 リスナー管理

ユーザー定義型のリスナー管理共通インタフェースを実装した。(表 11) 高機能版では、C++言語の仮想関数によるインタフェース、C 言語版ではコールバック関数により同機能を実装した。

表 11: リスナー管理機能一覧

No	機能	高機能版	軽量版	補足
1	リスナー管理	○	○	

5.2.2. UPnP 基本機能部

UPnP 基本機能部は、基本機能部のライブラリを基盤に実装され、UPnP で用いられる各種ネットワークプロトコルを実装した。本機能部は、今後の開発状況により仕様変更の可能性もあり、基本的にユーザー公開の対象とはしない。

5.2.2.2 HTTP クライアント/サーバ

HTTP クライアント/サーバ機能を実装した。(表 12) 高機能版は HTTP/1.1(RFC2616)、軽量版は HTTP/1.0(RFC1945)に準じた機能を対象とした。ただし、軽量版も、一部必須ヘッダ(Content-Length, Server 等)については HTTP/1.1(RFC2616)に準じた。

表 12 : HTTP クライアント/サーバ機能一覧

No	機能	高機能版	軽量版	補足
1	HTTP ヘッダ管理部	○	○	
2	HTTP クライアント部	○	○	
3	HTTP サーバ部	○	○	

5.2.2.3 XML 処理

XML データを入力とし、独自形式の XML メモリ構造を構築するパーサ機能を実装した。(表 13) 高機能版/軽量版共に XML パーサとしては Expat を標準とし、libxml2 をコンパイルオプションで選択可能とした。高機能版では更に Xerces を XML パーサとして選択可能である。

表 13 : XML 処理機能一覧

No	機能	高機能版	軽量版	補足
1	XML ノード処理	○	○	
2	XML 属性管理	○	○	
3	XML パーサ部	○	○	

5.2.2.4 SOAP 処理

SOAP メッセージ処理機能、XML での HTTP リクエスト/レスポンスを解析する機能を実装した。(表 14) SOAP メッセージは通常の HTTP リクエスト/レスポンス経由で処理され、HTTP 拡張フレームワークには対応しておらず、HTTP クライアント/サーバ機能から派生して実装されている。

表 14 : SOAP 処理機能一覧

No	機能	高機能版	軽量版	補足
1	SOAP リクエスト処理	○	○	
2	SOAP レスポンスト処理	○	○	

5.2.2.5 SSDP 処理

UPnP デバイスの発見/応答に用いられている SSDP プロトコルのメッセージ処理機能を実装した。(表 15) SSDP メッセージ部は、HTTP ヘッダ部と同構造であり、HTTP クライアント/サーバ機能から派生して実装されている。

表 15 : SSDP 処理機能一覧

No	機能	高機能版	軽量版	補足
1	SSDP メッセージ送受信部	○	○	
2	SSDP プロトコル処理部	○	○	

5.2.2.6 GENA 処理

UPnP デバイスのイベント管理に用いられている GENA プロトコルのメッセージ処理機能を実装した。(表 16) GENA メッセージ部は、HTTP ヘッダ部と同構造であり、HTTP クライアント/サーバ機能から派生して実装されている。

表 16 : GENA 処理機能一覧

No	機能	高機能版	軽量版	補足
1	GENA クライアント部	○	○	
2	GENA サーバ部	○	○	

5.2.3. UPnP デバイス管理部

UPnP 全般に関する機能を実装した。本機能は上記で提供される全機能を用いて実装される。本機能部は、ユーザー公開の対象とし、各種ドキュメント整備の対象とした。

5.2.3.2 デバイス管理

UPnP デバイスを構成する、デバイス管理、サービス管理、アクション管理、イベント管理の各種機能を実装した。(表 17) 本機能は UPnP サービスを提供するサーバ側の機能であり、クライアント側となる UPnP コントロールポイント管理機能の基盤ともなる。

表 17 : デバイス管理機能一覧

No	機能	高機能版	軽量版	補足
1	デバイス管理	○	○	
2	サービス管理	○	○	
3	アクション管理	○	○	
4	イベント管理	○	○	

デバイス管理/サービス管理は、UPnP デバイス/サービスのユーザー側からの定義および取得、デバイス開始/停止/検索時等の SSDP 経由の通知/応答他を自動的に管理する機能である。

アクション機能は SOAP 経由、イベント管理は GENA 経由のコントロールポイント側からの要求/応答を管理する部分で、主要な部分は自動的に管理されるが、アプリケーションに応じたユーザー任意のリスナー関数が設定可能である。

5.2.3.2 コントロールポイント管理

同一ネットワークの UPnP デバイスを検索し管理する UPnP コントロールポイントの各種機能を実装した。(表 18) 本機能は UPnP サービスを提供するサーバに対するクライアント側の機能である。

表 18 : コントロールポイント機能一覧

No	機能	高機能版	軽量版	補足
1	コントロールポイント管理	○	○	
2	デバイス検索	○	○	
3	デバイス管理	○	○	

コントロールポイント管理は UPnP コントロールポイント側からの定義および取得、デバイス検索は SSDP 経由でデバイスを検索、デバイス管理は検索されたデバイスを管理する機能である。

軽量版に関しては、本機能はコンパイルオプションにより除外の対象とした。UPnP コントロールポイントの機能が開発者は、本コンパイルオプションの指定により、ライブラリサイズ軽量化の選択が可能である。

5.2.4. 軽量版

軽量版は、高機能版の経験を基にした C 言語による新規開発となり、軽量かつ高速性を主眼として設計した。

実施計画当初、開発期間および組み込み系プラットフォームにおける省サイズ/省メモリに重点を置いていた関係上、UPnP の全機能を実装の範囲としない方針ではあったが、UPnP 相互接続性向上のため全機能を実装する事とした。

5.2.4.1 オブジェクト指向

高機能版である「CyberLink for C++」および Java 版である「CyberLink for Java」と同等のインタフェースの、C 言語版実装を目的とし、新規ユーザーが利用および、既存高機能版ユーザーが軽量版に移行する際にも、直感的に本ライブラリを利用できるよう配慮した。

隠蔽に関しては、本開発対象機能をレイヤ別処理別に分類し、その機能毎に専用の構造体およびアクセス関数を提供し、ユーザーに直接データ構造を意識させないよう配慮した。隠蔽に関する詳細は、後術の各機能詳細分類および命名規則にて補足する。

派生に関しては、構造体の先頭を一致させることにより実現し、オブジェクト指向および省メモリの観点で効率的な設計を実施した。派生に関しての詳細は、後術の省メモリに関する項目で補足する。

多様性(ポリモフィズム)に関しては、構造体オブジェクトに関数ポインタを保持し C++言語や Java 言語と同様な書式インタフェースの採用も検討したが、一般的な C 言語書式とは馴染まないと判断し、リスナー関数の登録レベルに留めた。

5.2.4.2 移植性

各プラットフォームへの移植性を確保するため特定のライブラリに依存しないよう最初公約数的な設計にて進めた。結果的に軽量版に関しては、主に組み込み系はプラットフォームによりファイル構造が規定されていない場合もありえる為、ファイル構造は本開発の対象外とした。

プログラムソースレベルでは ANSI C(ISO/IEC 9899:1990)準拠を前提とし、const 修飾子は関数返り値などキャストの必要性がある場合も考慮し、関数定義では用いなかった。

5.2.4.3 モジュール構造

ユーザー側で、組み込みたい機能を取捨選択できるよう軽量版に関してはコンパイルオプションによるモジュール指定を可能とした。主目的は、ユーザー選択による不要機能の削除およびオブジェクトサイズの調整である。

UPnP デバイス管理機能/コントロールポイント管理機能部について、後者が必要でない開発者がコンパイルオプションにより、ライブラリサイズ軽量化を目的に選択可能とした。また UPnP デバイス管理機能についても、相互接続性に問題がでないと判断した、UPnP イベント管理機能についても同様に各機能をコンパイルオプションにより選択可能とした。(表 19)

表 19 : モジュールコンパイルオプション

コンパイルオプション	機能
CG_UPNP_NOUSE_CONTROLPOINT	UPnP コントロールポイント管理機能無効化
CG_UPNP_NOUSE_SUBSCRIPTION	UPnP イベント管理機能無効化
CG_UPNP_NOUSE_ACTIONCTRL	UPnP アクション管理機能無効化(Control 関連)
CG_UPNP_NOUSE_QUERYCTRL	UPnP アクション管理機能無効化(Query 関連)

このモジュール構造によるコンパイルオプション機能の提供により、開発者が最適な機能およびオブジェクトサイズを選択することが可能となる。

5.2.4.4 省メモリ

実行時のオーバヘッドを考慮し、最低限のメモリ操作で処理を実装する。特に多段スタックのように各レイヤで同一または変換可能なデータは可能な限り、ZeroCopy により実装した。ただし、処理効率上、異なるデータ構造が必要となる場合には、データ構造解析および変換後、元となるデータ領域は速やかに解放するよう配慮
した。

5.2.4.5 省サイズ

最低限の処理によるコードサイズに配慮し設計した。具体的には、C 言語レベルでの継承実装を、親子関係にあるオブジェクトの構造体先頭メンバを同一にすることで実装した。この実装により、派生クラスは基底クラスのアクセス関数を利用でき、大幅なライブラリの省サイズ化も実現できた。例として、継承関係にある、基底クラスであるリスト構造体(CgList)および、派生関係にあるリスト構造

で管理される HTTP ヘッダ構造体(CgHttpHeader)の定義を以下に示す。(表 20)

表 20 : 構造体派生例

基底クラス (リスト構造管理構造体)	派生クラス (HTTP ヘッダ構造体)
typedef struct _CgList { **BOOL headFlag;** **struct _CgList *prev;** **struct _CgList *next;** } CgList;	typedef struct _CgHttpHeader { **BOOL headFlag;** **struct _CgHttpHeader *prev;** **struct _CgHttpHeader *next;** CgString *name; CgString *value; } CgHttpHeader, CgHttpHeaderList;

また、C 言語のマクロ機能で実装できる関数については、可能な限りマクロによる展開を実施した。しかしながら、関数表記に比べ可読性およびドキュメント作成の容易さが犠牲となる弊害があった。例として、上記リスト構造体(CgList)の次メンバアクセス関数(cg_list_next)および、その派生クラスである HTTP ヘッダ構造体での同関数のマクロ定義を以下に示す。(表 21)

表 21 : マクロ展開例

基底クラス関数	CgList *cg_list_next(CgList *list);
派生クラスマクロ	#define cg_http_header_next(header) (CgHttpHeader *)cg_list_next((CgList *)header)

当初は、オブジェクトサイズを 100KB 程度とした目標設定に向けて設計および実装作業を進めたが、UPnP 全機能を実装の対象とした事もあり、最終的なオブジェクトサイズはモジュール選択(コンパイルオプション)により、227KB～178KB の範囲となった。以下に代表的なコンパイルオプション指定時の、オブジェクトサイズを示す。(表 22) オブジェクトサイズは FedoraCore1 GNU GCC v3.3.2 でサイズ最適化オプション(Os)指定時の結果である。

表 22 : コンパイルオプション指定時オブジェクトサイズ

コンパイルオプション	○:有効 ×:無効				
CG_UPNP_NOUSE_CONTROLPOINT	×	○	○	○	○
CG_UPNP_NOUSE_SUBSCRIPTION	×	×	○	○	○
CG_UPNP_NOUSE_ACTIONCTRL	×	×	×	×	○
CG_UPNP_NOUSE_QUERYCTRL	×	×	×	○	○
オブジェクトサイズ	227KB	211KB	194KB	187KB	178KB

上記オプションの全有効時は、実際には UPnP デバイスとしてのサービスが全く提供できず実用的なものではないが、コンパイルオプション指定による最小オブジェクトサイズの参考値として表記した。

最終的なオブジェクトサイズについては、UPnP は実装範囲が広く、他商用ベンダーの C 言語による実装調査の結果[31]でも UPnP デバイス機能のみ(コントロールポイント機能含まず)の実装で 200KB 程度と表記されており、ある程度妥当な結果と判断している。本成果物で UPnP デバイス機能のみのコンパイルオプションは CG_UPNP_NOUSE_CONTROLPOINT のみ指定時のオブジェクトサイズ 211KB に相当する。

5.2.4.6 堅牢性

基本的には堅牢な関数(例: ×:strcpy, ○strncpy)を利用するが、プラットフォームにより利用できない場合も多いため移植性を兼ねて堅牢なラッパー関数で吸収した(例:cg_strncpy)。同様に仕様的に特定条件下の動作に問題がある関数(例:strncpy)も吸収し、それらの堅牢なラッパー関数群により、各機能を実装した。例として、文字列関連でラップした関数群を以下に示す。(表 23)

表 23 : ラッパー関数例

ラッパー関数	対象関数
int cg_strlen(char *str)	size_t strlen(const char *str)
char *cg_strdup(char *str)	char *strdup(char *str)
char *cg_strncpy(char *str1, char *str2, size_t cnt)	char *strncpy(char *dst, const char *src, size_t len)

5.2.4.7 命名規則

オブジェクト指向的な観点および、C 言語には C++の名前空間(NameSpace)や Java のパッケージのような言語処理サポートが他ライブラリと関数名衝突の可能性があるため、以下の命名規則を用いた。(表 24)

表 24 : 命名規則

対象	命名規則
定数	「CG」から開始し、センテンス毎にアンダーバー(_)で区切る 例 : CG_UPNP_XML_DECLARATION
クラス (構造体)	「Cg」から開始し、小文字および大文字の混合で、先頭およびセンテンス毎に大文字 <クラス> ::= cg_<クラスカテゴリ>_<サブクラスカテゴリ> 例 : CgUpnpDevice
関数	「cg_」から開始し小文字でセンテンス毎にアンダーバー(_)で区切る <関数名> ::= cg_<クラスカテゴリ>_<サブクラスカテゴリ>_<操作識別子> 例 : CgUpnpDevice *cg_upnp_device_new(); void cg_upnp_device_delete(CgUpnpDevice *dev);

また上記は、T-Engine ミドルウェア配布形式である T-Format も参考し、本成果物の T-Engine ミドルウェアとしての流通に影響がないよう配慮した。ただし、本命名規則により T-Engine ベンダーコードは「cg」となるが、これは T-Engine フォーラムから正式に割り当てられたものではなく、将来を含め他ベンダーと競合の可能性はある。

5.2.5. 高機能版

高機能版に関しては、従来からオープンソースで公開していたプロジェクトであり、基本的な設計および機能は維持し、組み込み系プラットフォームでの動作を可能にするため、軽量化および新規プラットフォームへの移植作業を実施した。

5.2.5.1 XML パーサ軽量化

従来から Xerces を標準 XML パーサとしてきたが、組み込み系プラットフォーム対応のため Expat を対応パーサに追加した。また libxml2 についても本成果物のオープンソース公開後に、ユーザーからの貢献があり対応パーサとして追加した。

5.2.5.2 プラットフォーム依存部移植

組み込み系プラットフォームを対象に、以下の基本機能部機能を各プラットフォーム固有のライブラリを用いて移植した。(表 25)

表 25 : 組み込み系依存移植対象機能

大区分	機能	T-Engine	μITRON	BTRON
基本機能部	ネットワーク処理	○	○	△
	ファイル管理	×	×	○
	タスク管理	○	○	○
	時刻管理	○	○	○

ファイル管理機能については、T-Engine/μITRON では標準でファイルシステムが規定されていない為、移植の対象外とした。またネットワーク処理機能については、BTRON ではプラットフォーム機能不備のためマルチキャスト機能未実装、μITRON ではコード上での移植処理は完了したが、一部試験作業を継続中である。

5.2.5.3 軽量化

主として文字列ストリーム(sstream)系の処理を、通常の文字列追加処理に変更し、一部処理の軽量化を実施した。この作業は、BTRON 環境で出力文字ストリーム(ostringstream)がサポートされていない現状にも起因した。なお高機能版は STL をデータ構造クラスに多用しているが、組み込み環境によっては開発環境標準のコンパイラで STL 対応していない場合もある。STL 自身には基本的に機種依存はなく移植も容易で、STLPort[17]等の軽量 STL も存在しているが、T-Engine で標準ストリーム(iostream)系の処理速度が遅い現象が確認できており、更なる軽量化を追求できる可能性が残された。

5.2.2.4 ファイル依存関数の変更

従来の高機能版では、UPnP デバイス管理部の UPnP デバイス/サービス定義は XML ファイル指定により実装されていた。組み込み系プラットフォームに依ってはファイルシステムが定義されていない場合もあり、同機能をファイル指定だけではなく、メモリ上に確保された XML データ領域から指定できるよう拡張を実施した。

6. 成果公開

6.1. 試験公開

7 月下旬から、既存の高機能版ユーザーからの要望で本成果物の試験公開を実施した。(表 26) 各ユーザー共通の背景とし、より軽量の UPnP プロトコルスタックを切望しており、有用なバグフィックスならび機能追加の貢献を頂いた。特に、本成果物に対して貢献度が大きかったものは、WindowsCE プラットフォーム対応および XML パーサ libxml2 対応追加である。

表 26 : 試験公開対象者一覧

代表者	所属組織	貢献内容	適用範囲
Aapo Makela	Nokia[18]	・軽量版評価/機能追加提案 ・高機能版 libxml2 対応 ・軽量版 libxml2 対応	社内研究用
Johannes Gutlebe	CERN[19]	・軽量版評価 ・高機能版評価/バグフィックス	研究用 (既存サービス発見プロトコルの UPnP への移行基盤として)
Theo Beisch	GMX[20]	・軽量版評価/バグフィックス ・Windows CE 移植確認作業協力	案件受注用プロトタイプ作成
Rosfran Borges	未確認	・軽量版評価 ・高機能版評価/バグフィックス	研究用(DBus[21]との連携)

WindowsCE 対応は、当初高機能版を WindowsCE に移植を試みていたユーザーと協力作業を実施した。結果的には高機能版の移植に関しては STL 関連問題で移植作業は休止状態となったが、軽量版に関して移植作業は順調に完了し本成果物に反映できた。また、同ユーザーからは Philips 製等の各種市販デバイスの動作確認および、その不具合報告またはパッチを貢献して頂いた。

libxml2 対応は、ユーザーから高機能版の libxml2 対応の貢献があり、筆者にて同貢献を参考に軽量版対応作業を実施した。ただし、筆者にて軽量版対応後、同ユーザーより libxml2 対応の貢献の連絡があり、ユーザーと作業が重複してしまった一面もあった。この XML パーサ対応は、同様に本成果物に反映できた。

6.2. 本公開

8 月下旬、個人管理サイトにて、軽量版のオープンソースでの一般公開を開始した。(表 27) 公開ライセンスは両者共、商用での利用および広範囲の本成果物の利用を考慮し BSD を選択した。高機能版に関しては、開発期間中にオープンソースユーザーからのパッチ等の貢献が多数あったため、これらを本成果物に取り込んでからの一般公開を予定している。

表 27 : 本公開情報

種別	名称	公開ライセンス	公開バージョン	URL
高機能版	CyberLink for C++	BSD	1.71	http://www.cybergarage.org/net/upnp/c/index.html
軽量版	CyberLink for C	BSD	1.0	http://www.cybergarage.org/net/upnp/cc/index.html

公開の告知は、UPnP フォーラムのメーリングリストおよび個人ブログサイト[22]で実施した。同メーリングリストは、UPnP フォーラムが主催であり、UPnP 開発者を含めた UPnP 関連でも世界最大のユーザ団体であり、告知には最適な場所である。

また配布ファイルのダウンロードサイトとして SourceForge[23]を利用した。SourceForge はオープンソース公開サイトとして世界最大、無料かつ各種統計情報が参照できるため、本成果物の結果を管理しやすいサイトして選択した。以下に、軽量版公開初日の結果を示す。(表 28) 軽量版公開時の全体の登録プロジェクト数は 10,5600 件であり、Rank はその登録プロジェクト数内の順位、Total Page はダウンロードファイルページの参照数、Download は実際のダウンロード数である。

表 28 : 軽量版公開初日結果

種別	日付	Rank	Total Page	Download
高機能版	2005/08/25	3554	72	24
軽量版	2005/08/25	6621	258	68

軽量版に関しては、公開初日で順位(Rank)こそ高機能版に及ばなかったものの、UPnP フォーラムのメーリングリストメンバーからの注目度は高く、公開ページや開発系ドキュメントの整備と合わせ、少なくとも高機能版と同等の注目度のあるプロジェクトとして発展させていきたい。今後、高機能版を上回る有力なプロジェクトとなる可能性を秘めている。

7. 開発成果の特徴

7.1. 利用対象者

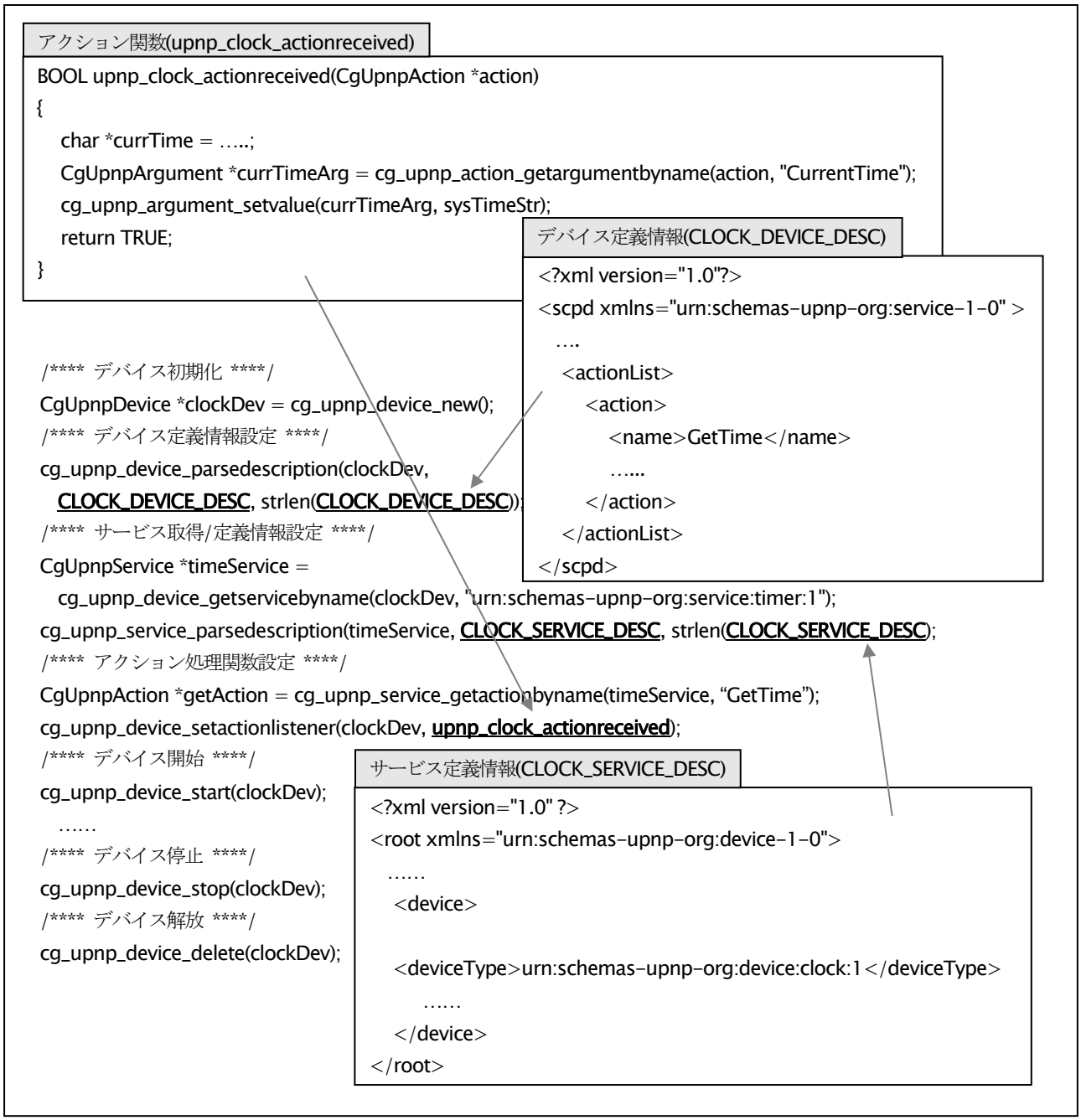
本開発成果の利用対象者は、UPnP による家庭内ネットワークまたは組み込み機器の相互接続環境構築の必要性がある開発者が対象である。商用/非商用を問わず、本成果物を利用した UPnP 関連アプリケーションが今後公開されることを期待している。

7.2. オブジェクト指向

高機能版および軽量版共にオブジェクト指向で設計されており、UPnP デバイス開発に必要となる全ての機能は抽象化されている。その為、UPnP に関する技術知識がない開発者を含め、本成果物により容易に UPnP デバイス開発が可能である。また、既存アプリケーションを UPnP 対応にする事も同様に容易なため、本成果物が将来的に UPnP 対応アプリケーションの普及に貢献する可能性もある。

本成果物を利用した場合 UPnP デバイス開発で必要となる作業は、UPnP デバイス/サービス定義情報およびサービスを実行するアクション関数の作成のみである。以下に、軽量版(C 言語)にて UPnP デバイスとしての時計アプリケーションを作成するサンプルを示す。(図 3)

図 3 : 軽量版時計 UPnP デバイスアプリケーションサンプル



7.3. 多言語対応

本成果物の対象となった高機能版/軽量版は開発言語として C++/C を対象としているが、別途 Java 版も同様にオープンソースにて公開中である。開発者は、開発プラットフォームや技術的好み等を考慮し、開発言語として C++/C/Java を選択可能である。

また、これらの UPnP プロトコルスタックは、開発言語は異なるが同一の基本設計にて開発されており、例えば高機能版の C++言語で開発してきたユーザーが、軽量版の C 言語の開発に際に移行する際の、移行障壁は限りなく低い。以下に、各プログラミング言語版の UPnP デバイス生成および開始/停止までのサンプルを示すが、処理的には同一ものであり本質的な差異はなく、共通の設計基盤で開発が可能である。(表 29)

表 29：プログラミング言語による UPnP デバイス実装ソース比較

高機能版(C++)	高機能版(Java)	軽量版(言語)

using namespace CyberLink;
Device *dev = new Device();
dev->loadDescriptoin(….);
dev->setActionListener(dev);
dev->setQueryListener(dev);
dev->start();
……
dev->stop();
	import org.cybergarage.upnp;
	Device dev = new Device();
	dev.loadDescripton(……);
	dev. setActionListener(dev);
	dev. setQueryListener(dev);
	dev.start();
	……
	dev.stop();
		CgUpnpDevice *dev = cg_upnp_device_new();
		cg_upnp_device_parsedescription(dev, ….);
		cg_upnp_device_setactionlistener(dev, …);
		cg_upnp_device_setquerylistener(dev, …);
		cg_upnp_device_start(dev);
		……
		cg_upnp_device_stop(dev);

7.4. 軽量性

組み込み系である T-Engine/μITRON を対象プラットフォームとし、高機能版を含めて軽量性を重視した。高機能版は、従来版を XML パーサ変更等の作業で従来版を軽量化した。新規開発の軽量版については C 言語レベルのオブジェクト指向設計や最適化作業により、オブジェクトサイズが 220KB と他商用ベンダー実装結果からも、ある程度遜色のない結果が得られた。

7.5. 移植性

一般的な Windows(WIN32)や Linux プラットフォームだけではなく、Unix 全般および組み込み系の T-Engine/μITRON を含む、幅広いプラットフォームを対象としている。

他新規プラットフォームに関しても、高機能版/軽量版共に、プラットフォーム差異は基本機能部階層で全て吸収しており、基本機能部のみ移植すれば容易に対応できる。基本機能部の上位階層の UPnP 基本管理部/UPnP デバイス管理部については、プラットフォーム依存コードは存在しないため、基本的には移植の必要性はない。

高機能版に関しては C++言語および STL を多用しているため、プラットフォームのコンパイラ品質によっては移植に労力が必要な場合もあるが、軽量版には ANSI C 準拠/依存ライブラリは XML パーサのみであるため、ソースコードとしても軽量でより移植に適している。

7.6. オープンソース

オープンソースの一般的な特徴については割愛するが、公開ライセンスとして BSD を選択しているため、商用/非商用を問わず同ライセンス下で自由に活動できる。GPL/LGPL での公開も検討したが、本成果物の利用ユーザー/改修状況の把握に強制力を持つ反面、商用利用に関しては不都合な側面も多々あると判断し、同ライセンスの選択に至った。

本成果物には、UPnP の基盤を支える HTTP クライアント/サーバや SOAP 処理等の実装、幅広いプラットフォームへの移植結果が含まれており、本成果物のソースを参照する事による教育用的効果もあると考えている。

また、開発中の試験公開結果に示すとおり、オープンソースユーザーからの協力貢献があり、本成果物にも実施計画書にはない追加機能およびプラットフォームに対応を追加できた。今後も、優秀な技術者の貢献を期待できる点も大きな魅力である。今後もオープンソース公開における利点を享受し、本成果物の発展および完成度を向上につなげていきたい。

7.7. 利用実績

筆者自身、本成果物を用いた UPnP/AV サーバ/クライアントである「CyberMediaGate for C++」等の開発者でもあるが、本成果物は、既に様々なユーザーによるアプリケーション開発に利用されている。実例として、本成果物開発中に高機能 C++/Java 版ユーザーから連絡のあった、本成果物を利用したアプリケーションの一例を以下に示す。

図 4 は、本成果物高機能版および「CyberMediaGate for C++」用いてオープンソース系動画再生クライアントである VLC[24] 最新版に UPnP AV 拡張および、組み込みデバイス向けに移植を実施した例である。XML パーサについては、本成果物の軽量化の特徴である Expat を選択している。

図 4 : UPnP AV Client for Embedded Device[25]

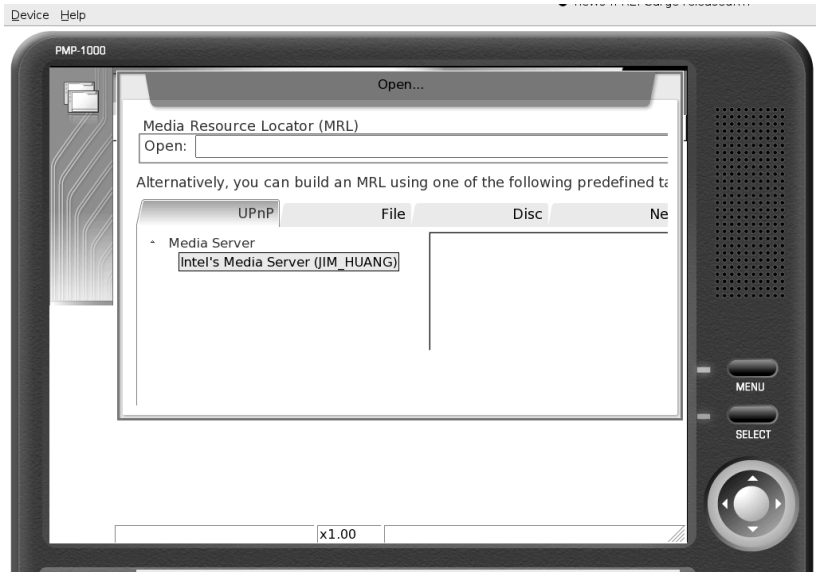
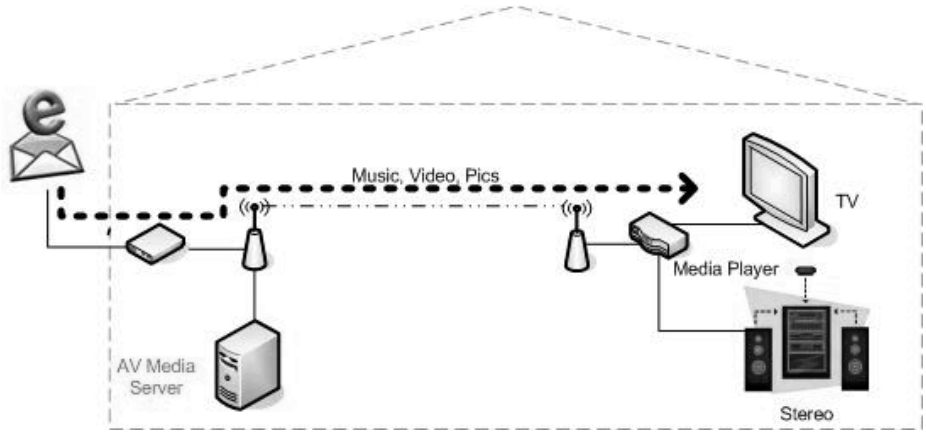


図 5 は本成果物高機能版の姉妹版となる「CyberLink for Java」および「CyberMediaGate for Java」を用いた例ではあるが、電子メールで送信されたコンテンツを UPnP AV コンテンツとして配信できる既存にはない特色があるアプリケーションである。この電子メールベースの試みは公開後の評判が良好で、今後は「CyberMediaGate for Java」の UPnP AV フレームワークの拡張を含めて、共同で作業を継続していく予定である。

図 5 : Email-based UPnP Media Server[26]



軽量版に関しては、本公開後の動向はこれからではあるが、既に試験公開中に様々なアプリケーション開発のベースとして本成果物を利用して頂いている実績はある。特に CERN では、本成果物の完成度が評価され、研究用のサービス発見プロトコルを既存の SLP(Service Location Protocol)から UPnP ベースへの移行を本格的に検討中との連絡もあり、高機能版と共に今後の普及に期待をしている。

7.8. 他社製品との比較

UPnP プロトコルスタックとしては、UPnP 仕様自体は個人/企業問わず一般に公開されているが、その標準リファレンス実装は提供されていない。UPnP プロトコルスタックの提供企業は年々増加しているが、現状でもそのほとんどが商用である。以下に、UPnP フォーラムに SDK [27]として登録されている UPnP プロトコルスタックの比較を示す。(表 30)

表 30 : UPnP プロトコルスタック製品比較表

プロダクト名 (ベンダー名)	プラットフォーム				開発言語				ライセンス	ソース公開
	WIN32	Linux	Java	T-Engine	C	C++	Java	.NET		
CyberLink	○	○	○	○	○	○	○	×	BSD	○
RomLink[28]	○	○	×	×	○	×	×	×	商用	○
aveLink[29]	○	○	○	×	○	×	○	×	商用	×
EBSnet[30]	-	-	-	-	○	×	×	×	商用	-
Jungo[31]	×	○	×	×	○	×	×	×	商用	×
Intel[32]	×	○	×	×	○	×	×	×	BSD	○
Microsoft[33]	○	×	×	×	○	-	-	○	商用	○
ProtoSys[34]	-	-	○	-	○	×	○	×	商用	-
Siemens[35]	○	×	×	×	×	○	○	×	商用	×
Wipo[36]	-	-	-	-	○	×	○	×	商用	-

*) 調査時アクセス不能なプロダクトは除外。調査結果として不明な項目については「-」表記とした。

他製品と比較して、プラットフォームおよび開発言語の幅広く対応しているのが、本成果物の特徴である。また T-Engine に対応したものとしては、おそらく本成果物が最初と思われる。

オープンソースとしては本成果物を除き、過去に Intel が Linux を対象に提供したものしか存在せず、現在そのメンテナンスは休止状態である。

8. 今後の課題、展望

8.1. オープンソース活動の継続

本成果物に関しては、基本設計は完了しており今後も大幅な変更は予定していないが、オープンソースによる公開を継続し、更なるユーザーニーズの調査およびバグフィックス等の作業により、完成度向上を目指している。

8.2. プラットフォーム拡充

本プロジェクトでは組み込み機器として、T-Engine/μITRON を対象としたが、近々には EmbeddedLinux、将来的にはワンチップマイコンに代表されるような、より低価格で処理性能の低いプラットフォームへの対応を予定している。個人レベルでは、現状の組み込み機器は金額的な面を含め開発環境構築の負担が大きく、このような一般的に入手しやすい組み込み系プラットフォームに対応できれば、個人の趣味レベルであっても、ユビキタス機器開発基盤として本成果物の適用範囲が広がるものと期待している。

μITRON に関しては、現状一部未実装/動作確認中であり、今後も PM からの支援を受けての移植作業を継続していく。また T-Engine/BTON に関しても、開発元の対応が得られれば、動作プラットフォームを拡充する意味でも、速やかに対応して行きたい。

8.3. 多言語対応

本成果物を、より多くの開発者に利用してもらう為、現状対応している C++, C, Java のプログラミング言語に加え、特定プラットフォームでメジャーな言語あるいはスクリプト系言語への対応を予定している。具体的には MacOSX の Object-C および Ruby を対象言語として、軽量版をベースとした対応を予定している。特にスクリプト言語への対応は、下記情報家電ビジュアル言語の実行エンジンを想定しており、その布石としたい。

8.4. 高速化

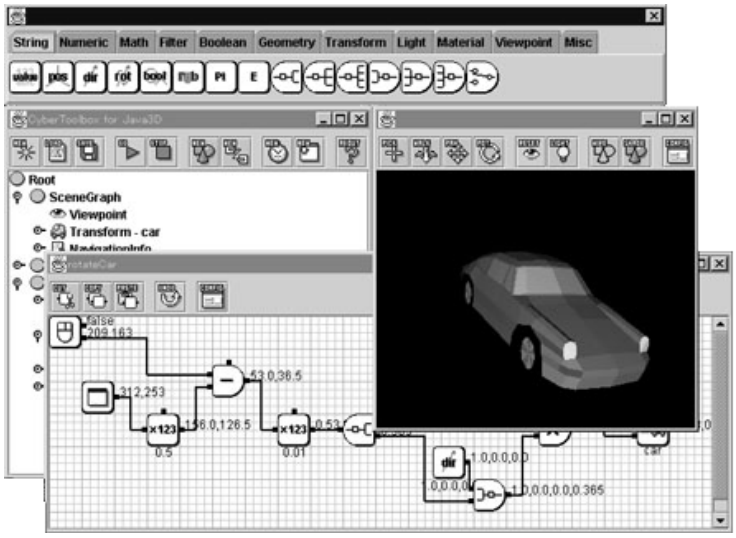
軽量版に関しては、機能的な実装および組み込み系プラットフォームを対象とした軽量化については完了したが、実行速度については更なる追及をしたい。現状でも開発プラットフォーム上でUPnPアクション処理時のSOAP処理等、実行に時間がかかる処理が確認できており、組み込み系プラットフォーム上での特性を踏まえてのプロファイルによりボトルネックを特定し、更なる処理速度向上を狙って行きたい。

8.5. 情報家電ビジュアル言語

家庭内の情報家電を相互接続して定型処理(例: ドアが開けばエアコンが動作)等の設定が、初心者でも簡単に設定できるようなビジュアル言語作成を構想している。

筆者は以前３Ｄコンピュータグラフィックスの分野で、コンテンツ制作用ビジュアル言語を開発した経緯（図 6)があり、それを踏まえての開発となる。基本構想としては、データフローにオブジェクト指向要素を取り入れ、情報家電向けビジュアル言語を、本成果物を基盤として開発を目指す。またオブジェクト指向ビジュアル言語単体としての開発/公開も視野に入れている。

図 6　: 3D コンテンツ制作ビジュアル言語 (CyberToolbox for Java3D)



9. 実施計画書内容との相違点

9.1. UPnP 全機能実装

実施計画当初、開発期間の要因もあり、軽量版に関してはUPnPの全機能を実装の範囲としないことで、組み込み系プラットフォームにおける省サイズ/省メモリに重点を置いていた。

しかしながらUPnP規格を更に厳密に補完する意味合いのIntel社のNMPR(Networked Media Products Requirements)[37]やDLNA等、UPnPデバイスの実装推奨規格が普及しつつある現状もあり、UPnPの機能実装を省略することは、本成果物が他UPnPデバイス/コントロールポイントとの相互接続性に問題が発生する可能性があった。その為、軽量版に関しても高機能版同様、UPnP全機能を実装の対象とし、その範囲内で軽量化の設計を実施した。

基本的に高機能版および軽量版はUPnP機能を全実装しているが、その動作基盤となる軽量版のHTTP仕様についてはHTTPv1.0をベースとした。NMPR等の実装推奨規格を含めUPnP仕様範囲ではHTTPv1.1への対応は必須項目ではなく、HTTPv.1.1ではオーバースペックな面もある。その為、相互接続性向上目的でHTTPv1.1での一部必須ヘッダ(Content-Length, Server等)へ対応除き、軽量化の観点でHTTPv1.0をベースに選択した。

9.2. オープンソース公開成果反映

軽量版に関して、実施計画書にはなかったマイクロソフト系の組み込み系プラットフォームである WindowsCE を対応プラットフォームとして本成果物に反映できた。また当初作業計画では、対応 XML パーサは高機能版が Xerces および Expat、軽量版が Expat のみであったが、libxml2 を高機能版/軽量版共、同様に本成果物に反映できた。これは、本成果物をオープンソースとして試験公開した結果によるものであり、ユーザーによる多大な貢献に感謝したい。

9.3. BTRON 事前調査不備

BTRON に関しては、BTRON 仕様および超漢字 4 で TCP/IP プロトコル実装の一部機能不備が判明したため、対象プラットフォームとしての優先度を下げた。ただし、マルチキャスト関連部以外のネットワーク機能、その他タスクや同期管理等の移植可能な機能について作業は完了させ、将来的な拡張に備えるものとした。

今後、BTRON(超漢字)でのマルチキャストサポートに関しては、坂村 PM から開発元であるパーソナルメディア株式会社に働きかけて頂くとの支援もあり、プラットフォーム対応後は速やかに対応予定である。移植に関する工数は 20 ステップ数程度である。

9.4. μITRON 移植作業遅延

μITRON に関しては、高機能版/軽量版共にプラットフォーム依存部を含む全ソースの移植作業は、ネットワーク関連部に関して動作確認作業が残っている段階である。具体的には、検証プラットフォームで高機能版/軽量版共に移植およびコンパイル作業およびタスクや同期制御等の基本的な移植動作確認は完了したが、開発環境のネットワーク関連の動作に問題があり、動作検証が遅れている段階である。

本プラットフォーム移植作業については、本委託契約期間中の作業完了を目標とするが、今後も継続開発し時期公開時にはリリースに含む予定としている。進捗遅延の原因としては、T-Engine 移植工数増加およびネットワーク部移植工数の増加(μITRON 特有の固定リソースや TCP/UDP 関数仕様)の影響もあるが、プラットフォームの選定/開発構築にベンダー支援状況を含め影響が大きかった。開発プラットフォームの再選定を含め、今後も PM の支援を得ながら本プラットフォームの移植作業を継続予定である。

10. おわりに

まずは、本事業により個人レベルでの組み込み系プラットフォーム開発、UPnP プロトコルスタックの新規設計の機会を与えてくれた、本プロジェクトおよび採択して頂いた坂村健東京大学大学院教授に感謝したい。

初めての組み込み系プラットフォームでの開発、久々に C 言語の制約範囲でライブラリを構築していく感覚等、本プロジェクトを通じて、新規分野へ挑戦する醍醐味およびプログラミングの魅力につき再認識させられる、非常に有意義な開発期間を過ごさせて頂いた。

またプロジェクトマネージャの坂村健東京大学大学院教授、YRP ユビキタスネットワーキング研究所の越塚登/新堂克徳両氏には本プロジェクトにおいて様々なご指導を頂いた。また、Aapo Makela、Johannes Gutlebe、Theo Beisch、Rosfran Borges には本成果物の機能および完成度向上に多大な貢献を頂いた。ここに記して感謝したい。

参考文献

[1] UPnP Forum – http://www.upnp.org/
[2] DLNA – http://www.dlna.org/
[3] MediaServer V 1.0 and MediaRenderer V 1.0 – http://www.upnp.org/standardizeddcps/mediaserver.asp
[4] CyberLink for C++ – http://www.cybergarage.org/net/upnp/cc/index.html
[5] CyberLink for Java ++ – http://www.cybergarage.org/net/upnp/java/index.html
[6] CyberMediaGate for C++ – http://www.cybergarage.org/net/cmgate/cc/index.html
[7] CyberMediaGate for Java – http://www.cybergarage.org/net/cmgate/cc/index.html
[8] CyberX3D for C++ – http://www.cybergarage.org/vrml/cx3d/cx3dcc/index.html
[9] bison – http://www.gnu.org/software/bison/bison.html
[10] flex – http://www.gnu.org/software/flex/
[11] Xerces C++ – http://xml.apache.org/xerces-c/
[12] Expat – http://expat.sourceforge.net
[13] libxml – http://xmlsoft.org/
[14] CppUnit – http://sourceforoge/projects/cppunit/
[15] CppUnit-x – http://sourceforge.jp/projects/cppunit-x/
[16] CUnit – http://sourceforge.net/projects/cunit/
[17] STLport – http://www.stlport.org/
[18] Nokia – http://www.nokia.com/
[19] CERN – www.cern.ch
[20] GMX – http://www.gmx.net/
[21] DBus – http://www.freedesktop.org/Software/dbus
[22] CyberGarage Blog – http://www.cybergarage.org/blog/skonnoblog.html
[23] SourceForge – http://sourceforge.net
[24] VLC – http://www.videolan.org/vlc/
[25] UPnP patch for VLC – http://jserv.sayya.org/upnp/
[26] Email-based UPnP Media Server – http://avmedia.org/avmedia
[27] UPnP Forum Sortware Development Kits (SDKs) – http://www.upnp.org/resources/sdks.asp
[28] RomPlug – http://www.allegrosoft.com/romplug.html
[29] aveLink – http://www.avelink.com/
[30] EBSnet – http://www.ebsnetinc.com/embedded-software-documents.php?id=72
[31] Jungo – http://www.jungo.com/components_upnp.html
[32] libupnp – http://upnp.sourceforge.net
[33] Platform SDK – www.microsoft.com/msdownload/platformsdk/
[34] ProtoSys – http://www.protosys.com/
[35] Simense – http://www.plug-n-play-technologies.com/
[36] Wipro – http://www.wipro.com/homenet
[37] Intel Developers Network for Digital Home – http://www.intel.com/technology/dhdevnet/

関連 Web サイト

CyberGarage – http://www.cybergarage.org